KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Rational Krylov methods for nonlinear eigenvalue problems

Roel Van Beeumen

Dissertation presented in partial
fulfillment of the requirements for the
degree of Doctor in Engineering Science

April 2015

Rational Krylov methods for nonlinear eigenvalue problems

Roel VAN BEEUMEN

Examination committee:
Prof. dr. ir. P. Verbaeten, chair
Prof. dr. ir. W. Michiels, supervisor
Prof. dr. ir. K. Meerbergen, supervisor
Prof. dr. R. Vandebril
Prof. dr. ir. G. Lombaert
Prof. dr. V. Mehrmann
(TU Berlin)
Prof. dr. F. Tisseur
(The University of Manchester)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering Science

April 2015

Preface

At the end of my PhD, it is time to look back on the past 3.5 years and to thank all the people who have supported me during this project.

First of all I would like to express my deepest appreciation to my supervisors Prof. W. Michiels and Prof. K. Meerbergen. I am grateful to have had the opportunity to do my PhD under their supervision. My collaboration with Karl already started in the summer of 2009 when he became my master's thesis supervisor. When he asked me a couple of months later if I would like to start a PhD in his research group, I immediately accepted his invitation. However, I first wanted to finish my archaeology studies. After finishing my fourth summer campaign in Sagalassos, I started my PhD in October 2011 and Wim joined the team. Karl and Wim, I would like to thank you for your guidance, support and advice, for our extensive weekly meetings, and for carefully proofreading all my papers, also in the weekends. I have really appreciated your efforts to be available despite of your busy schedules. Even when I unexpectedly entered your offices I was always welcome. Finally, I also want to thank you for releasing me every summer for the excavations in Turkey.

I would like to thank all jury members for accepting to take part in the examination committee, for carefully reading my dissertation text, and for the useful suggestions and comments. I thank Prof. P. Verbaeten for chairing the committee, Prof. R. Vandebril and Prof. G. Lombaert for their assistance during my PhD, and Prof. V. Mehrmann and Prof. F. Tisseur for the extensive discussion during the preliminary defense.

During my PhD, I also had the opportunity to travel and to spend some time abroad. In the spring of 2013, I had the luck to be supported by a FWO travel grant to visit the research group of Prof. F. Tisseur and Prof. N. Higham in Manchester. In particular, I want to thank Dr. S. Güttel for his company and collaboration. In addition I also thank all the colleagues over there such as Amal, Bahar, Edvin, James, Leo, Lijing, Mary, Meisam, Natasa, Ramaseshan, Sam, Sophia, Vanni, Vedran, and Yuji for the interesting discussions, but also for our drinks in the Sandbar and the Zoo. Next, I would like to thank Prof. E. Mengi and Prof. E. Jarlebring for hosting me during short research visits in winter of

2013–2014 to Istanbul and Stockholm, respectively. Despite of the cold weather I got the possibility to see something more of those beautiful cities.

Research is team work. Therefore, I also thank all my other collaborators Cedric, Dries, Katrien, Massimo, Matthias, Natalia, Nicola, and William.

I have had the pleasure of sharing office 03.26 with two fantastic office mates. Annelies and Jerzy, I thank you for our long and sometimes philosophic discussions. Sorry for all the P&O boxes stored in our office. Furthermore, I would like to thank my other colleagues at the department Albert-Jan, Ben, Benjamin, Cindy, Clara, Dan, Dirk, Dong, Gowri, Heba, Hendrik, Jakub, Jochen, Katrijn, Koen, Korneel, Laurent, Maryam, Matthias, Md, Micol, Nele, Nick, Nico, Peter, Piers, Pieter, Przemyslaw, Roel, Sam, Sathish, Stephanie, Thomas, Ward, Yaidel, Yao, and Zifan. I have enjoyed the many interesting discussions and the numerous enjoyable dinners with all of you. Nick, thanks a lot for your company on several conference trips of which the first one was unexpectedly extended and for our daily tea/coffee breaks.

I am also grateful to our secretary and in particular to Margot for arranging my flights, conference hotels and much more. Due to your effort, I was well shielded from a lot of administration, such that I could focus on my research. I would also like to thank our system group and Dirk for their assistance.

My passion for mathematics, science and technology originates from much earlier times. Therefore, I would also like to thank all my mathematics teachers in secondary school and in particular Mr. F. Gyssels and Mr. M. Verhees for encouraging me to start engineering studies.

Apart from my PhD research, archaeology dominated my summer months. Thanks to Prof. M. Waelkens and Prof. J. Poblome of the Sagalassos Archaeological Research Project, I could, and still can, recharge every year my batteries on the excavations of Sagalassos in the mountains of Turkey. However, my thoughts to my research were always latently present. It is even so that my first implementation of the rational Krylov method was made in Sagalassos. Also many thanks to all my friends of the excavation team for the great times together. In particular, I would like to thank Inge for our collaboration on the Upper Agora and the Domestic Area. Furthermore, thank you Joeri for all our evening discussions after the long and sometimes exhausting working days.

I would like to thank my music teachers Ann, Christian, Geert, and Veerle for providing me musical distraction during my PhD. I would also like to apologize for all my excuses not being able to practice enough.

Last but not least, I would like to thank my family: my parents for their unconditional support and my grandfather for always giving me down to earth advice. For sure I may not forget my *favourite* cousins Lynsey and Kim for the student life together and all our dinners.

Roel Van Beeumen
Heverlee, April 2015

Abstract

Eigenvalue problems arise in all fields of science and engineering. The mathematical properties and numerical solution methods for standard, linear eigenvalue problems are well understood. However, recent advances in several application areas resulted in a new type of eigenvalue problem, i.e., the nonlinear eigenvalue problem which exhibits nonlinearity in the eigenvalue parameter.

The goal of this thesis is to develop new rational Krylov methods for solving both small-scale and large-scale nonlinear eigenvalue problems. Firstly, by using polynomial and rational interpolation of the matrix-valued functions, we obtain methods which are globally convergent inside the region of interest. Secondly, linearization of the corresponding polynomial and rational eigenvalue problems results in linear pencils. Thirdly, the exploitation of the special structure of the linearization pencils and possibly a low rank structure results in efficient and reliable software which is publicly available.

We propose the Compact Rational Krylov (CORK) method as a generic class of numerical methods for solving nonlinear eigenvalue problems. CORK is characterized by a uniform and simple representation of structured linearization pencils. The structure of these linearization pencils is fully exploited and the subspace is represented in a compact form. Consequently, we are able to solve problems of high dimension and high degree in an efficient and reliable way.

The family of CORK methods has a lot of flexibility for solving the nonlinear eigenvalue problem. We discuss three particular types of CORK methods. The first one is the Newton Rational Krylov method which makes use of dynamic polynomial interpolation. The second one is the Fully Rational Krylov method which uses rational interpolation and has three viable variants: a static, dynamic, and hybrid variant. The third one is the Infinite Arnoldi method which uses an operator setting to solve the nonlinear eigenvalue problem. Finally, the proposed methods are used to solve applications from mechanical engineering, quantum physics, and civil engineering which were not solved earlier with the same efficiency and reliability.

Samenvatting

Eigenwaardenproblemen komen voor in alle wetenschapsdomeinen en ingenieursdisciplines. Voor het standaard lineaire eigenwaardenprobleem zijn de wiskundige eigenschappen en numerieke oplossingsmethodes goed gekend. Nieuwe ontwikkelingen in verschillende toepassingsdomeinen hebben echter aanleiding gegeven tot een nieuw type, namelijk het niet-lineaire eigenwaardenprobleem dat op een niet-lineaire manier afhangt van de eigenwaardenparameter.

Het doel van dit proefschrift is het ontwikkelen van nieuwe rationale Krylov methodes voor het oplossen van zowel kleinschalige als grootschalige niet-lineaire eigenwaardenproblemen. Ten eerste, door gebruik te maken van veelterm- en rationale interpolatie van de matrixfuncties bekomen we methodes die globaal convergeren in het gebied onder beschouwing. Ten tweede resulteert linearisatie van de overeenkomstige veelterm- en rationale eigenwaardenproblemen in lineaire matrixpencils. Ten derde zorgt de uitbuiting van de speciale structuur van de linearisatiepencils en mogelijk ook een lage rangstructuur voor efficiënte en betrouwbare software die publiek beschikbaar is.

We stellen de Compacte Rationale Krylov (CORK) methode voor als een algemeen raamwerk van numerieke methodes voor het oplossen van niet-lineaire eigenwaardenproblemen. CORK wordt gekarakteriseerd door een uniforme en eenvoudige representatie van de gestructureerde linearisatiepencils. We buiten de structuur van deze linearisatiepencils volledig uit en stellen de deelruimte op compacte wijze voor. Bijgevolg zijn we in staat om problemen van hoge dimensie en hoge graad op een efficiënte en betrouwbare manier op te lossen.

De familie van CORK methodes beschikt ook over veel flexibiliteit voor het oplossen van het niet-lineaire eigenwaardenprobleem. We behandelen drie specifieke types CORK methodes. De eerste is de “Newton Rational Krylov” methode en maakt gebruik van dynamische veelterminterpolatie. De tweede is de “Fully Rational Krylov” methode en gebruikt rationale interpolatie. We onderscheiden drie bruikbare varianten: een statische, dynamische en hybride variant. De derde is de “Infinite Arnoldi” methode en gebruikt een operatorsformulering voor het oplossen van het niet-lineaire eigenwaardenprobleem. Tot slot gebruiken we de voorgestelde methodes voor het oplossen van problemen

uit de mechanica, kwantumfysica en bouwkunde die nooit eerder opgelost waren met dezelfde efficiëntie en betrouwbaarheid.

Contents

Abbreviations

CORK	Compact rational Krylov
DEP	Delay eigenvalue problem
GEP	Generalized linear eigenvalue problem
LEP	Linear eigenvalue problem
NLEP	Nonlinear eigenvalue problem
PEP	Polynomial eigenvalue problem
QEP	Quadratic eigenvalue problem
REP	Rational eigenvalue problem

List of Symbols

Mathematical notation

$\alpha, \beta, \ldots$	Scalars
$x, y, \ldots$	Vectors
$\mathbf{x}, \mathbf{y}, \ldots$	Block vectors
$A, B, \ldots$	Matrices
$\mathbf{A}, \mathbf{B}, \ldots$	Block matrices
$\Omega, \Sigma, \ldots$	Sets
$\mathcal{A}, \mathcal{B}, \ldots$	Operators

Special constants

i	Imaginary unit
I	Identity matrix
I_n	Identity matrix of size $n \times n$
$I_{n \times m}$	Identity matrix of size $n \times m$
$0_{n \times m}$	All-zero matrix of size $n \times m$

Sets

$\mathbb{N}$	Set of natural numbers
$\mathbb{R}$	Set of real numbers
$\mathbb{C}$	Set of complex numbers

Functions

$f(\lambda)$	Scalar function
$f(M)$	Matrix function
$A(\lambda)$	Matrix-valued function

Matrix operations and properties

A^{-1}	Inverse of A
A^T	Transpose of A
A^*	Conjugate transpose of A
$\det(A)$	Determinant of A
$\operatorname{rank}(A)$	Rank of A
$\operatorname{range}(A)$	Range of A
$\operatorname{diag}(\alpha_1, \ldots, \alpha_n)$	Diagonal matrix with arguments on its diagonal

Scalar product and norms

$\langle \cdot, \cdot \rangle$	Scalar product
$\lvert \cdot \rvert$	Absolute value
$\lVert \cdot \rVert$	Euclidian norm
$\lVert \cdot \rVert_1$	1-norm
$\lVert \cdot \rVert_2$	2-norm

Other

$\overline{\cdot}$	Complex conjugate
$\operatorname{Re}(\cdot)$	Real part
$\operatorname{Im}(\cdot)$	Imaginary part
$\otimes$	Kronecker product
e_k	kth unit vector
$x^{[k]}$	kth block component of $\mathbf{x}$
$O(\cdot)$	Big O
$\operatorname{span}\{x_1, \ldots, x_n\}$	Linear span of the argument vectors

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

Eigenvalue problems lie at the heart of computational science and engineering, including the earthquake response of a building or bridge, stability analysis of cars and airplanes, Google's page rank algorithm, population growth, and the energy levels of molecules in quantum physics. The mathematical properties and numerical solution methods for standard, linear eigenvalue problems are well understood.

Recent advances in several application areas, such as structural dynamics and sound and vibration analysis, have led to the emergence of new types of eigenvalue problems exhibiting nonlinearity in the eigenvalue parameter. We denote this type of nonlinear problem as the *nonlinear eigenvalue problem*, i.e, the linear system

$$A(\lambda)x = 0,$$

where A is a family of square matrices depending on a complex parameter λ. The parameters λ for which this system has a nontrivial solution are called *eigenvalues*. The corresponding solution vectors x are called *eigenvectors*.

This chapter is organized as follows. Firstly, we define the nonlinear eigenvalue problem in Section 1.1. We also discuss some special classes and applications of nonlinear eigenvalue problems, and give a brief overview of the state-of-the-art numerical methods for solving general nonlinear eigenvalue problems. Next, we discuss the motivation and objectives in Section 1.2. In Section 1.3 we review two iterative methods for solving large-scale standard linear eigenvalue problems. Finally, we give an overview of the thesis in Section 1.4.

1.1 Nonlinear eigenvalue problems

We define the nonlinear eigenvalue problem (NLEP) as follows

$$A(\lambda)x = 0, \tag{1.1}$$

where the *eigenvalues* $\lambda \in \Omega \subseteq \mathbb{C}$, the matrix-valued function $A : \Omega \to \mathbb{C}^{n\times n}$, and corresponding (right) *eigenvectors* $x \in \mathbb{C}^n \setminus \{0\}$. We assume that $A(\lambda)$ is analytic on Ω, except for possibly a countable number of isolated singularities in Ω. Solving this nonlinear eigenvalue problem concerns to find parameters λ for which the linear system (1.1) has nontrivial solutions x.

Remark that the nonlinear eigenvalue problem by definition is *linear* in the eigenvector x and only *nonlinear* in the eigenvalue λ. Nonlinearity in the eigenvector will not be considered in this thesis. In contrast to the standard eigenvalue problem, the nonlinear eigenvalue problem (1.1) can have an infinite number of eigenvalues and also the eigenvectors can be linearly dependent.

Firstly, we review in §1.1.1 several special classes of the nonlinear eigenvalue problem. Next, we describe in §1.1.2 four applications of nonlinear eigenvalue problems in totally different fields of science and engineering. Finally, we give a brief overview of existing numerical methods for solving nonlinear eigenvalue problems in §1.1.3.

1.1.1 Special classes

The matrix-valued function $A(\lambda)$ in (1.1) has the property that it can always be written in the following form

$$A(\lambda) = \sum_{i=1}^{m} B_i f_i(\lambda),$$

where $B_i \in \mathbb{C}^{n\times n}$ are constant matrices, f_i are scalar functions of λ, and $m \leq n^2$. However, in many applications $m \ll n^2$. We now briefly review the main special classes of the nonlinear eigenvalue problem.

Linear eigenvalue problem

The first class is the standard *linear* eigenvalue problem (LEP)

$$Ax = \lambda x,$$

where the matrix $A \in \mathbb{C}^{n\times n}$ and which can be written in the form (1.1) as

$$(A - \lambda I)x = 0.$$

This linear eigenvalue problem has been studied for decades in the field of numerical linear algebra. The power method [111], introduced by von Mises and Pollaczek–Geiringer in 1929, is a numerical algorithm for computing eigenvalues of a constant matrix. This method computes only one eigenvalue and the corresponding eigenvector at once and can converge very slowly.

Currently, the QR algorithm, independently introduced by Francis [35, 36] and Kublanovskaya [56] in the early 1960s, is one of the standard algorithms for computing all the eigenvalues of a matrix A. In case matrix A is large and sparse, iterative projection methods are used such as the Implicit Restarted Arnoldi algorithm [63]. This method computes only a small subset of the eigenvalues of the matrix A.

The linear eigenvalue problem is actually a special case of the generalized linear eigenvalue problem (GEP)

$$(A - \lambda B)x = 0,$$

where the matrices $A, B \in \mathbb{C}^{n\times n}$. Similar techniques as for the linear eigenvalue problem can be used for the generalized linear eigenvalue problem.

Quadratic eigenvalue problem

The second class is the *quadratic* eigenvalue problem (QEP)

$$(\lambda^2 M + \lambda C + K)x = 0,$$

where the matrices $M, C, K \in \mathbb{C}^{n\times n}$. This type of eigenvalue problems occurs in a wide range of vibration applications including dynamical analysis of structures and acoustics. In these cases M is the mass matrix, C the damping matrix, and K the stiffness matrix. For a general survey of the quadratic eigenvalue problem, we refer to [100] and the references therein.

The QUADEIG algorithm [45] can be used for computing all $2n$ eigenvalues of a quadratic eigenvalue problem. However, in several applications only the eigenvalues with smallest real parts are of interest. Therefore, linearization based Arnoldi algorithms were proposed, e.g., [12, 72].

Polynomial eigenvalue problem

Both the linear and quadratic eigenvalue problem are special cases of the *polynomial* eigenvalue problem (PEP)

$$P(\lambda)x = \left(\sum_{i=0}^{k} \lambda^i A_i\right) x = 0,$$

where the matrices $A_i \in \mathbb{C}^{n\times n}$. The standard approach for solving polynomial eigenvalue problems is via linearization [38], i.e., the reformulation into a larger generalized linear eigenvalue problem with the same eigenvalues. For more details, see, e.g., [69, 3].

Rational eigenvalue problem

The polynomial eigenvalue problem is again a special case of the more general *rational* eigenvalue problem (REP) [98]

$$Q(\lambda)x = \left(P(\lambda) + \sum_{i=1}^{k} \frac{s_i(\lambda)}{q_i(\lambda)} A_i\right) x = 0,$$

where $P(\lambda)$ is an $n \times n$ matrix polynomial, $s_i(\lambda), q_i(\lambda)$ are scalar polynomials, and the matrices $A_i \in \mathbb{C}^{n\times n}$. This type of eigenvalue problems occurs in a wide range of applications such as optimization of acoustic emissions of high speed trains, free vibration of plates with elastically attached masses, and vibration of fluid-solid structures. For an overview, see [98] and the references therein.

Delay eigenvalue problem

The final special class we will discuss here originates from the stability analysis of linear time-delay systems [74] described by the following delay-differential equation

$$\dot{x}(t) = A_0 x(t) + \sum_{i=1}^{k} A_i x(t - \tau_i),$$

where $x(t) \in \mathbb{R}$ is the state vector at time t, the matrices $A_i \in \mathbb{R}^{n\times n}$ for $i = 0, 1, \ldots, k$ are real, and $0 < \tau_1 < \tau_2 < \cdots < \tau_k$ represent the time-delays.

The substitution of a sample solution in the form of $e^{\lambda t}v$, with $v \in \mathbb{C}^n \setminus \{0\}$, gives rise to the *delay* eigenvalue problem (DEP)

$$A(\lambda)x := \left(\lambda I - A_0 - \sum_{i=1}^{k} e^{-\tau_i \lambda} A_i\right) x = 0.$$

Note that the delay eigenvalue problem has in general infinitely many eigenvalues. However, they have some nice and interesting properties. For more details and specialized solution methods, we refer to [74].

1.1.2 Applications

Nonlinear eigenvalue problems arise in all fields of science and engineering. Here, we give a selection of four applications from mechanical engineering, quantum physics, civil engineering, and electrical engineering. All these examples will be further used as examples in the next chapters of this thesis.

Viscoelastic damping for noise control

Viscoelastic damping technologies are currently widely used for vibration control and noise reduction in automotive, aerospace and aeronautics industries [80]. Constrained-layer damping, see Figure 1.1, which consists of a sandwich of two outer layers with a viscoelastic core, is one of the most commonly used viscoelastic treatments.

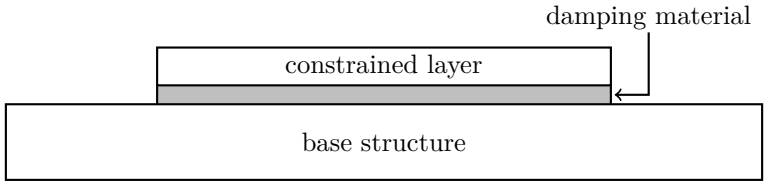


Figure 1.1: Constrained-layer damping.

The purpose of this sandwich structure arrangement is to increase the damping factor of the system subjected to bending forces. Furthermore, the inclusion of the viscoelastic material in the core of the sandwich structure will also contribute to reduce the total weight of the structure.

The use of viscoelastic material introduces nonlinear damping in the system, resulting in the following nonlinear eigenvalue problem

$$\left(K - \omega^2 M + f(\omega)C\right) x = 0, \tag{1.2}$$

where K, M, and C are constant matrices and the scalar function f depends in a nonlinear way on the angular frequency ω.

In Section 7.1, we will consider a clamped sandwich beam with viscoelastic core [71], denoted by `'sandwich beam'` problem, where the shear modulus of the sandwich structure is described by the four-parameter generalized Zener model. In this case, the function f in (1.2) is given by

$$f(\omega) = \frac{G_0 + G_\infty (i\omega\tau)^\alpha}{1 + (i\omega\tau)^\alpha},$$

where G_0 is the static shear modulus [Pa], G_∞ the asymptotic shear modulus [Pa], τ the relaxation time [s], and α the fractional parameter.

Bound states in semiconductor devices with contacts

The study of electronic transport in semiconductor devices requires the simultaneous solution of the Poisson, Schrödinger, and transport equations. When collisions only have a minor impact on the determination of the device current, the ballistic picture is satisfactory. Thus, the current can be determined by solving the single-particle Schrödinger equation with open transmitting boundary conditions [65].

Determining bound states in a semiconductor device with contacts [109] requires the computation of the solutions of the following Schrödinger eigenvalue problem

$$\left(-\nabla^2 + U(x)\right)\chi(x) = \lambda\chi(x), \tag{1.3}$$

where $U(x)$ is the potential energy. A one-dimensional system with a left and right contact is shown in Figure 1.2.

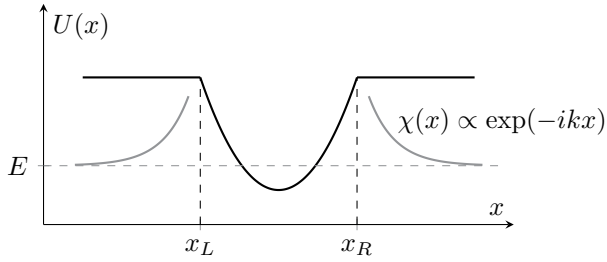


Figure 1.2: 1D potential with contacts starting at x_L and x_R.

Due to the nonlinear boundary conditions, the discretization of the Schrödinger equation (1.3) results in the nonlinear eigenvalue problem

$$\left(H - \lambda I + \Sigma(\lambda)\right)x = 0, \tag{1.4}$$

where H is a constant matrix and Σ depends in a nonlinear way on λ.

In the numerical experiments of §4.3.2 and in Section 7.2, we will consider the `'particle in a canyon'` problem of which we want to calculate the bound states and corresponding wave functions. In this case, the matrix-valued function $\Sigma(\lambda)$ in (1.4) is given by

$$\Sigma(\lambda) = \sum_{j=1}^{k} e^{i\sqrt{m(\lambda-\alpha_j)}} S_j,$$

where S_j are constant low-rank matrices, m the particle mass, and $\alpha_j \in \mathbb{R}$.

Surface waves in multilayered halfspaces

The spectral analysis of surface waves [77, 122] is used in civil engineering to determine the dynamic properties of a layered soil, which is represented by a multilayered halfspace as illustrated in Figure 1.3.

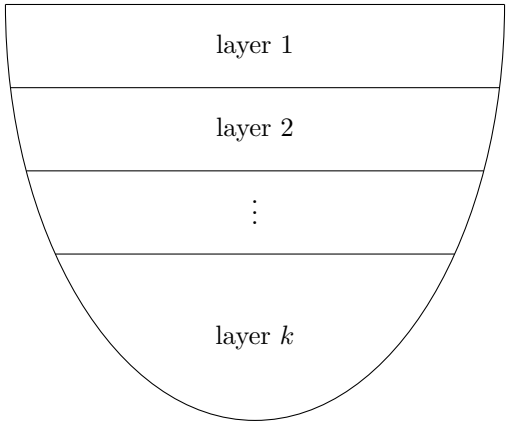


Figure 1.3: Multilayered halfspace.

Surface waves travel in the horizontal direction, along the free surface of the medium or along an interface between layers, and are evanescent in the vertical direction. Furthermore, they are dispersive, i.e., the phase velocity and the attenuation coefficient of a surface wave varies with the frequency. In a layered medium surface waves give rise to a nonlinear eigenvalue problem of the following form

$$K(k_x, \omega)x = 0,$$

where the stiffness matrix-valued function $K(k_x, \omega)$ depends in a nonlinear way on the horizontal wavenumber k_x and on the frequency ω.

In Section 7.3, we will consider a surface waves problem, denoted by `'surface waves'` problem, for a soil consisting of 3 different layers. In this case the matrix-value function $K(k_x, \omega)$ is given by

$$K(k_x, \omega) = \begin{bmatrix} K_1(k_x, \omega) & \\ & 0_{2\times 2} \end{bmatrix} + \begin{bmatrix} 0_{2\times 2} & \\ & K_2(k_x, \omega) \end{bmatrix} + \begin{bmatrix} 0_{4\times 4} & \\ & K_3(k_x, \omega) \end{bmatrix},$$

where K is of dimension 6 and K_1, K_2, and K_3 correspond to layer 1, layer 2, and layer 3, respectively.

Cavity design of a linear accelerator

Waveguides are used in accelerator design to introduce damping into the cavity to suppress higher order modes. For the corresponding electromagnetic modeling of waveguide loaded accelerator cavities, we need to solve Maxwell's equation

$$\nabla \times \left(\frac{1}{\mu}\nabla \times E\right) - \lambda \varepsilon E = 0 \qquad \text{in} \qquad \Omega,$$

with the following boundary conditions

$$\begin{aligned} \vec{n} \times E = 0 \qquad & \text{on} \qquad \Gamma_E, \\ \vec{n} \times \left(\frac{1}{\mu}\nabla \times E\right) = 0 \qquad & \text{on} \qquad \Gamma_M, \\ \vec{n} \times \left(\frac{1}{\mu}\nabla \times E\right) + i\sqrt{\lambda - \kappa_{c,j}^2}\,\vec{n} \times (\vec{n} \times E) = 0 \qquad & \text{on} \qquad \Gamma_j, \quad j = 1, \ldots, k, \end{aligned}$$

where E is the electric field, $\lambda = \omega^2/c^2$, ω is the wavenumber, c is the speed of light, ε is the relative permittivity, and μ is the relative permeability. Ω is the geometry of the cavity with the outward normal vector $\vec{n}$ on the boundary, Γ_E is the electric boundary with perfect conductor, Γ_M is the magnetic boundary with perfect insulator, and Γ_j are the waveguide boundaries. $\kappa_{c,j}$ are the cutoff values, i.e., the cutoff wavenumbers of the modes in the waveguide ports, and k is the number of waveguides. For more details see, e.g., [48, 68].

Finite element analysis of Maxwell's equation with nonlinear waveguide boundary conditions yields the following nonlinear eigenvalue problem

$$\left(K - \lambda M + i\sum_{j=1}^{k}\sqrt{\lambda - \kappa_{c,j}^2}W_j\right)x = 0,$$

where the stiffness matrix K, the mass matrix M, and the damping matrices W_j are constant real symmetric matrices. Furthermore, K is positive semi-definite and M is positive definite [61, 53].

Throughout this thesis we will use the `'gun'` problem [68], which is also part of the NLEVP collection [14], to compare the different methods. This problem models an open cavity with two waveguides resulting in the following nonlinear eigenvalue problem

$$\left(K - \lambda M + i\sqrt{\lambda - \kappa_{c,1}^2}W_1 + i\sqrt{\lambda - \kappa_{c,2}^2}W_2\right)x = 0, \tag{1.5}$$

where the cut-off values are $\kappa_{c,1} = 0$ and $\kappa_{c,2} = 108.8774$. The nonlinear eigenvalue problem (1.5) has become the last couple of years one of the most commonly used benchmark problems for testing nonlinear eigenvalue solvers because this problem is a large scale problem, fully nonlinear including branch cuts, and has low rank coefficient matrices W_1 and W_2.

1.1.3 Numerical methods

Over the last half century there is a vast literature on numerical methods for solving nonlinear eigenvalue problems. Although there exist specialized methods for special classes of nonlinear eigenvalue problems, we only consider here numerical methods for general nonlinear eigenvalue problems and distinguish the following four main classes. For a general overview see [73, 114, 31].

Newton-type methods

A first approach is to formulate the eigenvalue problems as a system of nonlinear equations and then solve it by variants of Newton's method or the inverse iteration method. All these methods are local methods, thus, are not guaranteed to converge. Moreover, these methods compute only one eigenvalue at once and reconvergence to an already computed eigenvalue cannot be prevented.

In the early literature, we find, for example, the generalized Rayleigh quotient iteration [60], the Newton's method and inverse iteration [4], QR-type methods [57, 58], the method of successive linear problems [82], and the residual inverse iteration method [78]. More recently, we have extensions capable of computing multiple eigenvalues, such as nonlinear Jacobi–Davidson methods [15, 113, 32], the nonlinear Arnoldi method [112], and the block Newton method [54].

Methods based on approximations of $A(\lambda)$

A second approach is based on polynomial approximation of the matrix-valued function $A(\lambda)$. This approach yields more globally convergent methods than Newton-type methods. However, the reliability of these methods heavily depends on the quality of the approximation of the nonlinear function.

In discretize-first methods, such as the Chebyshev interpolation method [33], $A(\lambda)$ is first approximated by an interpolating matrix polynomial. Next, the resulting polynomial eigenvalue problem is solved via linearization. On the other hand, in the Taylor–Arnoldi method [50] the degree of the polynomial approximation is increased in every iteration of the algorithm, yielding an increasing and better approximation of $A(\lambda)$ around a given shift σ.

Methods based on a linear operator eigenvalue problem

A third approach reformulates the nonlinear eigenvalue problem as a linear (infinite) dimensional operator eigenvalue problem. In the Infinite Arnoldi Method [52] this operator eigenvalue problem is solved by Arnoldi's method.

Methods based on contour integration

A fourth approach makes use of a contour integral reformulation of the nonlinear eigenvalue problem. All these methods compute the eigenvalues inside a given contour. See, e.g., [16, 7, 121]. For a connection with rational filters, see [104, 103].

1.2 Research aims

Nonlinear eigenvalue problems can have infinitely many eigenvalues, however, in practice only a relatively small number is of importance. Therefore, we consider in this thesis the problem of finding eigenvalues λ of the NLEP (1.1) in a specific region of interest $\Sigma \subset \Omega$.

1.2.1 Motivation

Although more and more nonlinear eigenvalue problems are appearing in different types of applications, efficient and reliable methods are sparse. The motivation of this thesis is threefold.

Firstly, Newton-type methods exhibit only local convergence and require good starting values in order to guarantee convergence.

Secondly, methods based on approximations of the nonlinear matrix-valued function are more globally convergent. However, for large-scale problems linearization yields very large problems. Especially for high order approximations, which are sometimes needed for accurate results, this technique is potentially inefficient.

Thirdly, contour integral methods exhibit global convergence inside the contour. However, these methods are rather expensive due to the high number of discretization points needed in order to accurately approximate the contour integrals.

1.2.2 Objectives

The main objectives of this thesis are to develop new rational Krylov methods for solving nonlinear eigenvalue problems that:

1. are **globally convergent** inside the region of interest;
2. are applicable to **both small-scale and large-scale** problems;
3. result in **efficient and reliable software**.

For globally convergent methods an accurate approximation, based on polynomial and rational interpolation of the matrix-valued function, is required. Linearization of polynomial and rational eigenvalue problems results in linear pencils and can be used for small- and large-scale nonlinear eigenvalue problems. Applicability to large-scale problems is achieved by rational Krylov methods which fully exploit the special structure and sparsity of these linearization pencils.

1.3 Iterative methods for linear eigenvalue problem

In order to make this thesis self-contained, we now briefly review two iterative methods for computing eigenvalues in a specific region of the complex plane of the linear eigenvalue problem (LEP)

$$Ax = \lambda x, \tag{1.6}$$

where $A \in \mathbb{C}^{n\times n}$, and of the generalized linear eigenvalue problem (GEP)

$$Ax = \lambda Bx, \tag{1.7}$$

where $A, B \in \mathbb{C}^{n\times n}$.

1.3.1 Arnoldi method

The Arnoldi method [6] is an orthogonal projection method for computing approximations of a subset of the eigenvalues and eigenvectors of a square matrix $A \in \mathbb{C}^{n\times n}$. The basic algorithm is outlined in Algorithm 1.1.

Algorithm 1.1: Arnoldi method

```
Choose vector v_1, where ||v_1|| = 1.
for j = 1, 2, ... do
    Compute: v̂ := A v_j.
    Orthogonalize: ṽ := v̂ − V_j h_j, where h_j = V_j^* v̂.
    Get new vector: v_{j+1} = ṽ / h_{j+1,j}, where h_{j+1,j} = ||ṽ||.
    Compute Ritz pairs: (λ_i, x_i) and test for convergence.
end
```

The Arnoldi method can be seen as a Gram–Schmidt orthogonalization process for building an orthogonal basis V_j of the Krylov subspace

$$\mathcal{K}_j(A, v_1) = \operatorname{span}\left\{v_1, Av_1, \ldots, A^{j-1}v_1\right\},$$

for the matrix A generated by the vector v_1. The approximate eigenvalues λ_i, called *Ritz values*, are computed as eigenvalues of the Hessenberg matrix

$$H_j = V_j^* A V_j,$$

obtained by the projection process onto $\mathcal{K}_j$. An approximate eigenvector, called *Ritz vector*, associated with a Ritz value λ_i is defined by $x_i = V_j y_i$, where y_i is an eigenvector of H_j associated with the eigenvalue λ_i.

By eliminating $\widehat{v}$ and $\widetilde{v}$ in the jth iteration of Algorithm 1.1 and combining all previous iterations, we get the Arnoldi recurrence relation

$$AV_j = V_{j+1}\underline{H}_j, \tag{1.8}$$

where $\underline{H}_j$ is an $(j+1) \times j$ upper Hessenberg matrix.

Orthogonalization

For the orthogonalization in step 3 in Algorithm 1.1, iterative Gram–Schmidt with reorthogonalization is used [26]. When $\widetilde{v}$ is computed, a test is performed to compare its norm to the norm of $\widehat{v}$. If $\|\widetilde{v}\|/\|\widehat{v}\|$ falls below a certain threshold, a second orthogonalization is made. For more details, see [62].

In each iteration step j, we assume that $h_{j+1,j} \neq 0$. Then, we call $\underline{H}_j$ *unreduced*. If $h_{j+1,j} = 0$, the range(V_j) is an invariant subspace and

$$AV_j = V_j H_j.$$

At this point, the Gram–Schmidt orthogonalization process fails and the algorithm breaks down. However, this only happens if and only if the starting vector v_1 is a linear combination of j eigenvectors. In this case the obtained eigenvalues and eigenvectors are exact.

Computing approximate eigenpairs

Approximations for the eigenvalues and corresponding eigenvectors of matrix A can, in each iteration j of Algorithm 1.1, be obtained from the small projected matrix $H_j \in \mathbb{C}^{j\times j}$.

Definition 1.1. *Let* (λ_i, s_i) *satisfy*

$$H_j s_i = \lambda_i s_i, \qquad s_i \neq 0. \tag{1.9}$$

Then we call $(\lambda_i, x_i := V_j s_i)$ *a Ritz pair of the matrix* A.

The accuracy of a Ritz pair (λ_i, x_i) is typically estimated by the residual norm $\|Ax_i - \lambda_i x_i\|$ and can be obtained as a by-product of the Arnoldi process. Let

$$r_i := Ax_i - \lambda_i x_i.$$

By using the substitution $x_i = V_j s_i$, the recurrence relation (1.8) and equation (1.9), respectively, yields

$$\begin{aligned} r_i &= AV_j s_i - \lambda_i V_j s_i, \\ &= V_{j+1}\underline{H}_j s_i - \lambda_i V_j s_i, \\ &= \begin{bmatrix} V_j & v_{j+1} \end{bmatrix} \begin{bmatrix} H_j s_i - \lambda_i s_i \\ h_{j+1,j}\left(e_j^T s_i\right) \end{bmatrix}, \\ &= \alpha_i v_{j+1}, \end{aligned}$$

where $\alpha_i := h_{j+1,j}\left(e_j^T s_i\right)$. Thus, a simple check for convergence of a Ritz pair in step 5 of Algorithm 1.2 results in

$$\left|\frac{\alpha_i}{\lambda_i}\right| \leq \varepsilon_{\text{tol}}, \tag{1.10}$$

where ε_{tol} is defined by the user.

Implicit restarting

The Arnoldi method is very effective for computing a small subset of the eigenvalues of a large sparse matrix. However, its convergence completely depends on the choice of the starting vector v_1 and there is no way to determine in advance how many iterations are needed to compute the eigenvalues of interest within a specific accuracy. Hence, the iteration count can become very large resulting in extensive storage requirements.

The first restarting technique, *explicitly restarting*, was introduced by Saad [86]. Explicit restarting means the replacement of the starting vector v_1 with an *improved* starting vector and then rerunning the algorithm with the new vector. The second restarting technique for the Arnoldi method, *implicitly restarting*, was introduced by Sorensen [96] and can be seen as follows. Suppose that after m iterations of the Arnoldi algorithm, we have a subspace V_{m+1} with corresponding Hessenberg matrix $\underline{H}_m$, which we want to reduce to a smaller subspace V_{k+1} and Hessenberg matrix $\underline{H}_k$ with $k < m$, such that they still satisfy the Arnoldi recurrence relation (1.8) and such that the eigenvalues of H_k are the k selected ones of H_m.

In order to carry out implicit restarting and locking of converged Ritz values easily, we work with a Krylov–Schur recurrence relation [97]. We start by computing a Schur decomposition of H_m

$$H_m = Z S_m Z^*,$$

where S_m is an upper quasitriangular matrix and $Z^*Z = ZZ^* = I$. Using this Schur decomposition, we can transform the Arnoldi recurrence relation into the following Krylov–Schur recurrence relation

$$AU_m = U_{m+1}\underline{S}_m,$$

where

$$U_{m+1} := \begin{bmatrix} V_m Z & v_{m+1} \end{bmatrix},$$

and

$$\underline{S}_m := \begin{bmatrix} S_m \\ h_{m+1}^* Z \end{bmatrix}.$$

Let S_m be an ordered Schur decomposition of H_m

$$S_m = Z^* H_m Z = \begin{bmatrix} Z_1 & Z_2 & Z_3 \end{bmatrix}^* H_m \begin{bmatrix} Z_1 & Z_2 & Z_3 \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{13} \\ & S_{22} & S_{23} \\ & & S_{33} \end{bmatrix},$$

where $S_{11} \in \mathbb{C}^{\ell\times\ell}$, $S_{22} \in \mathbb{C}^{(k-\ell)\times(k-\ell)}$, and $S_{33} \in \mathbb{C}^{(m-k)\times(m-k)}$ are upper quasi triangular matrices. The ordering is as follows: the eigenvalues of S_{11}, S_{22}, and S_{33} are, respectively, the very accurate Ritz values, the wanted but not yet converged Ritz values, and the unwanted Ritz values. Hence,

$$\underline{S}_m = \begin{bmatrix} S_{11} & S_{12} & S_{13} \\ & S_{22} & S_{23} \\ & & S_{33} \\ b_1^* & b_2^* & b_3^* \end{bmatrix}, \qquad U_{m+1} = \begin{bmatrix} U_1 & U_2 & U_3 & u_{m+1} \end{bmatrix}, \tag{1.11}$$

where $b_i^* = h_{m+1}^* Z_i$ for $i = 1, 2, 3$ and $\|b_1\|$ small. Note that, by using the ordered Schur decomposition of H_m, the relation

$$A \begin{bmatrix} U_1 & U_2 \end{bmatrix} = \begin{bmatrix} U_1 & U_2 & u_{m+1} \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ & S_{22} \\ b_1^* & b_2^* \end{bmatrix},$$

is also a Krylov–Schur decomposition. Thus, the purging problem can be solved by moving the unwanted Ritz values into the southeast corner of the matrix S_m and truncating the decomposition. Next the Arnoldi process is continued with v_{m+1} as next vector.

The use of the ordered Schur decomposition has also the advantage that deflation and locking of the converged Ritz pairs correspond to setting $b_1^* = 0$ in (1.11), yielding the following recurrence relation

$$A \begin{bmatrix} U_1 & U_2 \end{bmatrix} = \begin{bmatrix} U_1 & U_2 & u_{m+1} \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \\ 0 & b_2^* \end{bmatrix} + O(\|b_1\|).$$

Note that b_1 is a measure for the backward error of the corresponding eigenvalues of S_{11}. Hence, $\|b_1\|$ will be zero if the eigenvalues of S_{11} are exact. For more information we refer to [62, 63, 75, 97].

Spectral transformations

It is known that the standard Arnoldi method rapidly provides approximations to well-separated extremal eigenvalues [11]. In order to compute eigenvalues of the generalized linear eigenvalue problem (1.7) close to a shift $\sigma \in \mathbb{C}$, we perform the following shift-and-invert spectral transformation

$$(A - \sigma B)^{-1} B x = \theta x,$$

with

$$\theta = \frac{1}{\lambda - \sigma}.$$

The corresponding shift-and-invert Arnoldi method is outlined in Algorithm 1.2.

Algorithm 1.2: Shift-and-invert Arnoldi method

Choose shift $\sigma \in \mathbb{C}$ and vector v_1, where $\|v_1\| = 1$.
for $j = 1, 2, \ldots$ **do**
  Compute: $\widehat{v} := (A - \sigma B)^{-1} B v_j$.
  Orthogonalize: $\widetilde{v} := \widehat{v} - V_j h_j$, where $h_j = V_j^* \widehat{v}$.
  Get new vector: $v_{j+1} = \widetilde{v}/h_{j+1,j}$, where $h_{j+1,j} = \|\widetilde{v}\|$.
  Compute Ritzpairs: (λ_i, x_i) and test for convergence.
end

By elimination of $\widehat{v}$ and $\widetilde{v}$ in the jth iteration of Algorithm 1.2 and combining all previous iterations, we obtain again an Arnoldi recurrence relation

$$(A - \sigma B)^{-1} B V_j = V_{j+1} \underline{H}_j,$$

for the matrix $C := (A - \sigma B)^{-1} B$. Therefore, Algorithm 1.2 for the pencil (A, B) and with shift σ is equal to Algorithm 1.1 for the matrix C. In this case, the Ritz values are computed as follows

$$\lambda_i = \sigma + \frac{1}{\theta_i},$$

where θ_i are the eigenvalues of H_j. Note that the eigenvector x_i associated with θ_i in the transformed problem is also an eigenvector of the original problem (1.7) corresponding to λ_i.

1.3.2 Rational Krylov method

The rational Krylov method [83, 85], outlined in Algorithm 1.3, is a generalization of the shift-and-invert Arnoldi method for solving the generalized linear eigenvalue problem (1.7). There are two main differences between the two methods. Firstly, instead of a fixed shift for the Arnoldi method, the rational Krylov method allows to change the shift (or pole) at every iteration. Secondly, the rational Krylov method collects the information about the eigenvalues in a pair of Hessenberg matrices (K, H).

Algorithm 1.3: Rational Krylov method

Choose vector v_1, where $\|v_1\| = 1$.
for $j = 1, 2, \ldots$ **do**
  Choose shift: σ_j.
  Set continuation combination: t_j.
  Compute: $\widehat{v} := (A - \sigma_j B)^{-1} B V_j t_j$.
  Orthogonalize: $\widetilde{v} := \widehat{v} - V_j h_j$, where $h_j = V_j^* \widehat{v}$.
  Get new vector: $v_{j+1} = \widetilde{v}/h_{j+1,j}$, where $h_{j+1,j} = \|\widetilde{v}\|$.
  Compute Ritzpairs: (λ_i, x_i) and test for convergence.
end

The rational Krylov algorithm builds a subspace spanned by

$$v_1,\ (A - \sigma_1 B)^{-1} B v_1,\ (A - \sigma_2 B)^{-1} B v_2,\ \cdots$$

and by eliminating $\widehat{v}$ and $\widetilde{v}$ in the jth iteration in Algorithm 1.3 we get the relation

$$(A - \sigma_j B)^{-1} B V_j t_j = V_{j+1} \underline{h}_j, \tag{1.12}$$

where $\underline{h}_j = \begin{bmatrix} h_j^* & h_{j+1,j}^* \end{bmatrix}^*$. Combining all the previous iterations, we arrive at the recurrence relation of the rational Krylov method

$$A V_{j+1} \underline{H}_j = B V_{j+1} \underline{K}_j, \tag{1.13}$$

where $\underline{H}_j$ and $\underline{K}_j$ are two $(j+1) \times j$ upper Hessenberg matrices. The matrix $\underline{H}_j$ contains the coefficients of the Gram–Schmidt orthogonalization process and

$$\underline{K}_j := \underline{H}_j \operatorname{diag}(\sigma_1, \ldots, \sigma_j) + \underline{T}_j,$$

where $\underline{T}_j \in \mathbb{C}^{(j+1)\times j}$ is the upper triangular matrix built up from the continuation combinations $t_1, \ldots, t_j$. Note that from this definition, we can easily find that

$$\sigma_j = \frac{k_{j+1,j}}{h_{j+1,j}}.$$

The rational Krylov method Algorithm 1.3 is characterised by its matrices V, $\underline{H}$, and $\underline{K}$. In order to refer to the rational Krylov process (1.13), we introduce the term *RKS triple* [29].

Definition 1.2 (RKS triple)**.** *The triple* $(V_{j+1}, \underline{H}_j, \underline{K}_j)$ *with* $V_{j+1} \in \mathbb{C}^{dn\times(j+1)}$ *and* $\underline{H}_j, \underline{K}_j \in \mathbb{C}^{(j+1)\times j}$ *is called a RKS triple of order* j *for* (A, B)*, if*

1. *it satisfies the rational Krylov recurrence relation* (1.13)*,*
2. $\underline{K}_j$ *and* $\underline{H}_j$ *are upper Hessenberg matrices with* $\underline{H}_j$ *unreduced, and*
3. *none of the* $\sigma_i = k_{i+1,i}/h_{i+1,i}$*,* $i = 1, \ldots, j$ *is an eigenvalue of* (A, B)*.*

Continuation combination

If we use in every iteration of Algorithm 1.3 the same shift, $\sigma_j = \sigma_1$, and continue with the last vector, $t_j = e_j$, we get back the shifted and inverted Arnoldi relation

$$(A - \sigma_1 B)^{-1} B V_j = V_{j+1} \underline{H}_j,$$

However, the advantage of the rational Krylov method is that we can vary the shift in order to obtain good approximations to all the eigenvalues in a union of regions around the chosen shifts.

Similar to the shifted and inverted Arnoldi method, the rational Krylov method starts with a shift σ_1 and a starting vector v_1. But by changing the shift we need to avoid throwing away all the information gathered in the basis V constructed using the previous shift [11]. Therefore, for selecting the next continuation vector we replace the basis V by a new basis W. This basis spans the same subspace as V but can be interpreted as the orthogonal basis formed by the Arnoldi process with the new shift σ_2. The last vector of this new basis W is now taken as the next continuation vector.

Suppose that at iteration j of Algorithm 1.3 we change in step 2 the shift from σ_1 into σ_2. At this point, the recurrence relation

$$A V_j \underline{H}_{j-1} = B V_j \underline{K}_{j-1},$$

holds. Then, by subtracting $\sigma_2 B V_j \underline{H}_{j-1}$ from both sides, we get

$$(A - \sigma_2 B) V_j \underline{H}_{j-1} = B V_j (\underline{K}_{j-1} - \sigma_2 \underline{H}_{j-1}).$$

Next, by using the following QR decomposition

$$\underline{K}_{j-1} - \sigma_2 \underline{H}_{j-1} = Q_j \begin{bmatrix} R_{j-1} \\ 0 \end{bmatrix},$$

we obtain

$$(A - \sigma_2 B)^{-1} B V_j Q_{j-1} = V_j Q_j \underline{F}_{j-1}, \tag{1.14}$$

where $\underline{F}_{j-1} := Q_j^* \underline{H}_{j-1} R_{j-1}^{-1}$ is a full matrix. $\underline{F}_{j-1}$ can be transformed into upper Hessenberg form by applying the Householder algorithm from the bottom upwards [84] as follows

$$\underline{F}_{j-1} = \begin{bmatrix} P_{j-1} & 0 \\ 0 & 1 \end{bmatrix} \underline{\widetilde{H}}_{j-1} P_{j-1}^*, \tag{1.15}$$

with P_{j-1} an orthonormal matrix. Finally, substituting (1.15) into (1.14) yields the following shifted and inverted Arnoldi relation

$$(A - \sigma_2 B)^{-1} B W_{j-1} = W_j \underline{\widetilde{H}}_{j-1},$$

where $\underline{\widetilde{H}}_{j-1}$ is an upper Hessenberg matrix and

$$W_j := V_j Q_j \begin{bmatrix} P_{j-1} & 0 \\ 0 & 1 \end{bmatrix}.$$

The last vector of W_j is now taken as continuation vector corresponding to the new shift σ_2. This results in the continuation combination $t_j = Q_j e_j$. Thus, we select the continuation combination t_j in step 3 of Algorithm 1.3 as follows

$$t_j = \begin{cases} e_j, & \sigma_j = \sigma_{j-1}, \\ q_j = Q_j e_j, & \sigma_j \neq \sigma_{j-1}, \end{cases} \tag{1.16}$$

where $t_1 := e_1$ and Q_j is obtained from the QR factorization of

$$Q_j \underline{R}_{j-1} = \underline{K}_{j-1} - \sigma_j \underline{H}_{j-1}.$$

Computing approximate eigenpairs

Approximations for the eigenvalues and corresponding eigenvectors of matrix pencil (A, B) can, in each iteration j of Algorithm 1.3, be obtained from the $j \times j$ upper parts of the two Hessenberg matrices $\underline{H}_j$ and $\underline{K}_j$.

Definition 1.3. *Let* (λ_i, s_i) *satisfy*

$$K_j s_i = \lambda_i H_j s_i, \qquad s_i \neq 0.$$

Then we call (λ_i, x_i)*, where*

$$x_i := V_{j+1} \underline{H}_j s_i,$$

a Ritz pair of (A, B)*.*

As in the Arnoldi method, the convergence of a Ritz pair in step 7 of Algorithm 1.3 can be monitored by (1.10) with $\alpha_i := (k_{j+1,j} - \lambda_i h_{j+1,j})(e_j^T s_i)$.

Implicit restarting

Suppose that after m iterations of Algorithm 1.3, we have the RKS triple $(V_{m+1}, \underline{H}_m, \underline{K}_m)$. By implicitly restarting the rational Krylov method [29, 85] we can reduce this RKS triple to a smaller RKS triple $(V_{k+1}, \underline{H}_k, \underline{K}_k)$ with $k < m$.

Instead of making use of a Krylov–Schur recurrence relation in the Arnoldi method, we will use here a rational Krylov–Schur version. We start by computing a generalized Schur decomposition of H_m and K_m

$$H_m = Y S_m Z^*, \qquad K_m = Y T_m Z^*,$$

where S_m, T_m are upper quasitriangular matrices and Y, Z unitary. Substituting these generalized Schur decompositions in (1.13) and multiplying from the right by Z, we obtain the following rational Krylov–Schur recurrence relation

$$AU_{m+1}\underline{S}_m = BU_{m+1}\underline{T}_m,$$

where

$$U_{m+1} := \begin{bmatrix} V_m Y & v_{m+1} \end{bmatrix},$$

and

$$\underline{S}_m := \begin{bmatrix} S_m \\ h_{m+1}^* Z \end{bmatrix}, \qquad \underline{T}_m := \begin{bmatrix} T_m \\ k_{m+1}^* Z \end{bmatrix}.$$

Similarly as for the Arnoldi method, we reorder the eigenvalues of (T_m, S_m) and subdivide $\underline{S}_m$ and $\underline{T}_m$ as follows

$$\underline{S}_m = \begin{bmatrix} S_{11} & S_{12} & S_{13} \\ & S_{22} & S_{23} \\ & & S_{33} \\ b_1^* & b_2^* & b_3^* \end{bmatrix}, \qquad \underline{T}_m = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ & T_{22} & T_{23} \\ & & T_{33} \\ c_1^* & c_2^* & c_3^* \end{bmatrix},$$

where $S_{11}, T_{11} \in \mathbb{C}^{\ell \times \ell}$, $S_{22}, T_{22} \in \mathbb{C}^{(k-\ell)\times(k-\ell)}$, and $S_{33}, T_{33} \in \mathbb{C}^{(m-k)\times(m-k)}$ are upper quasi triangular matrices and $b_i^* = h_{m+1}^* Z_i$, $c_i^* = k_{m+1}^* Z_i$ for $i = 1, 2, 3$ assuming $\|b_1\|$ and $\|c_1\|$ small. Hence, purging the unwanted eigenvalues of (T_{33}, S_{33}) and locking the ones of (T_{11}, S_{11}), yields

$$AU_{k+1}\underline{S}_k = BU_{k+1}\underline{T}_k, \tag{1.17}$$

where

$$\underline{S}_k = \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \\ 0 & b_2^* \end{bmatrix}, \qquad \underline{T}_k = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \\ 0 & c_2^* \end{bmatrix},$$

and

$$U_{k+1} = \begin{bmatrix} U_1 & U_2 & u_{m+1} \end{bmatrix}.$$

Finally, note that the relation (1.17) is a standard rational Krylov recurrence relation of order k with $V_{k+1} := U_{k+1}$, $\underline{H}_k := \underline{S}_k$, and $\underline{K}_k := \underline{T}_k$.

1.4 Overview of the thesis

This thesis is organized as follows. An overview of the main chapters and its inner relations is presented in Figure 1.4.

Chapter 1 introduces this thesis by situating the research context and aims. It also contains a short overview of the standard iterative methods for solving large-scale linear eigenvalue problems.

Chapter 2 provides an overview of linearization of matrix polynomials in different bases. The linearization pencils and corresponding structured eigenvectors will form a basis for the rational Krylov methods presented in the next chapters.

This chapter is partially based on the paper [108] published in IMA Journal of Numerical Analysis.

Chapter 3 introduces the idea of dynamic polynomial interpolation where neither the interpolation points nor the degree of the interpolating matrix polynomial are fixed in advance. Both the Taylor–Arnoldi method and the Newton Rational Krylov method are discussed in this chapter.

This chapter is mainly based on the paper [106] published in SIAM Journal on Scientific Computing.

Chapter 4 extends the results of the previous two chapters to rational interpolation. It introduces the Fully Rational Krylov method of which we present a static, dynamic, and hybrid variant.

This chapter is mainly based on the paper [44] published in SIAM Journal on Scientific Computing.

Chapter 5 approaches the nonlinear eigenvalue problem from a different point of view. It is reformulated as an infinite dimensional linear operator eigenvalue problem.

This chapter contains the results of the paper [105] submitted to Numerical Linear Algebra with Applications.

Chapter 6 introduces a generic but simple representation of structured linearization pencils. The Compact Rational Krylov method maximally exploits its structure and unifies the Krylov methods of the previous chapters.

This chapter is integrally based on the paper [107] accepted for publication in SIAM Journal on Matrix Analysis and Applications.

Chapter 7 illustrates the software of the previous chapter for some real science and engineering applications.

This chapter contains the results of the paper [109] published in Journal of Computational Electronics.

Chapter 8 summarizes the main conclusions and contributions of this thesis. It also contains recommendations for further research.

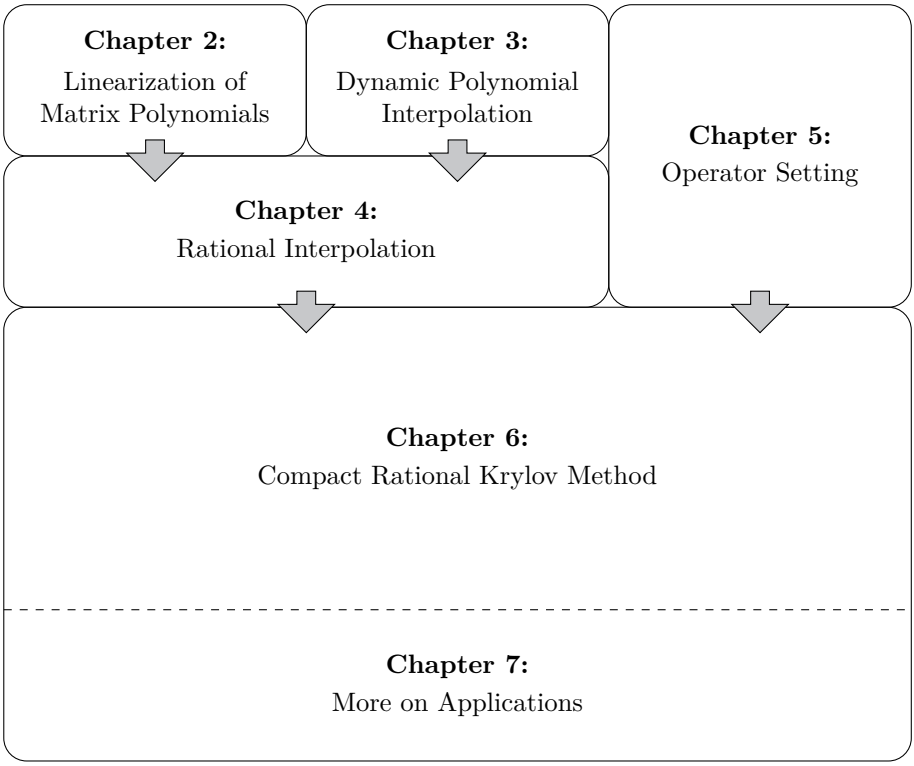


Figure 1.4: Overview of the thesis.

Chapter 2

Linearization of Matrix Polynomials

For many years already, linearization is the classical approach for solving polynomial eigenvalue problems. Firstly, the matrix polynomial is transformed into a larger linear pencil with the same eigenvalues. Next, a standard linear eigensolver can be applied to compute the eigenvalues of the polynomial eigenvalue problem. For any polynomial there exist infinitely many linearizations. However, companion-type linearizations are often used in practice.

This chapter is organized as follows. We start by introducing some basic definitions and notation in Section 2.1. Next, we review the matrix polynomial linearizations in monomial basis, Lagrange basis, orthogonal bases, and Newton basis in Section 2.2, Section 2.3, Section 2.4, and Section 2.5, respectively.

2.1 Definitions

We study linearizations of matrix polynomials

$$P(\lambda) = \sum_{i=0}^{d} A_i \lambda^i, \tag{2.1}$$

where $A_i \in \mathbb{C}^{n\times n}$ are constant matrices and d the *grade* [70]. In case $A_d \neq 0$, the grade is equal to the degree of $P(\lambda)$. For example, if $P(\lambda) \equiv A_0$, with A_0 a nonsingular matrix, then both the degree and grade of $P(\lambda)$ are 0, while for $P(\lambda) = 0\lambda^2 + 0\lambda + A_0$ the degree is still 0 but the grade is 2. We assume that $P(\lambda)$ is *regular*, i.e., $\det P(\lambda)$ does not vanish identically.

By linearization, we mean the conversion of $P(\lambda)$ into a larger linear matrix pencil $\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B}$ with the same eigenvalues [38, 69, 3]. For this purpose *unimodular* matrix polynomials are used, i.e., matrix polynomials $E(\lambda)$ such that $\det E(\lambda)$ is a nonzero constant and independent of λ. We start with the following definition of a (weak) linearization.

Definition 2.1 (Weak linearization [38])**.** *Let $P(\lambda)$ be an $n \times n$ matrix polynomial of grade d with $d \geq 1$. Then a pencil $\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B}$ with $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{dn \times dn}$ is called a linearization of $P(\lambda)$ if there exist unimodular matrix polynomials $E(\lambda)$, $F(\lambda)$ such that*

$$E(\lambda)\mathbf{L}(\lambda)F(\lambda) = \begin{bmatrix} P(\lambda) & 0 \\ 0 & I_{(d-1)n} \end{bmatrix}.$$

Thus, $\mathbf{L}(\lambda)$ is a linearization of $P(\lambda)$ if and only if the finite eigenvalues of $\mathbf{L}(\lambda)$, together with their partial multiplicities, coincide with those of $P(\lambda)$. Before stating the definition of a strong linearization, we introduce the reversal of a matrix polynomial in order to study eigenvalues of $P(\lambda)$ at ∞.

Definition 2.2 (Reversal of matrix polynomial)**.** *For a matrix polynomial $P(\lambda)$ of grade d, the reversal of $P(\lambda)$ is the polynomial $P^{\#}(\lambda) := \lambda^d P(\lambda^{-1})$.*

Note that the nonzero finite eigenvalues of $P^{\#}(\lambda)$ are the reciprocals of those of $P(\lambda)$ and that an eigenvalue at ∞ of $P(\lambda)$ corresponds to an eigenvalue 0 of the reversal polynomial $P^{\#}(\lambda)$.

Definition 2.3 (Strong linearization [37])**.** *Let $P(\lambda)$ be an $n \times n$ matrix polynomial of grade d with $d \geq 1$. If $\mathbf{L}(\lambda)$ is a linearization of $P(\lambda)$ and $\mathbf{L}^{\#}(\lambda)$ is a linearization of $P^{\#}(\lambda)$, then $\mathbf{L}(\lambda)$ is called a strong linearization of $P(\lambda)$.*

The additional property that $\mathbf{L}^{\#}(\lambda)$ is also a linearization of $P^{\#}(\lambda)$ ensures that for regular matrix polynomials the Jordan structure of the eigenvalue ∞ is preserved by strong linearizations.

2.2 Monomial basis

Let $P(\lambda)$ with $P : \mathbb{C} \to \mathbb{C}^{n \times n}$ be a matrix polynomial defined as in (2.1) with $A_d \neq 0$. One of the most commonly used linearizations for (2.1) is the companion form in the following theorem.

Theorem 2.4. *Let $P(\lambda)$ be an $n \times n$ matrix polynomial of degree d given in the form* (2.1)*. Then the $dn \times dn$ linear companion pencil*

$$\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B} = \begin{bmatrix} A_0 & A_1 & \cdots & A_{d-1} \\ & I & & \\ & & \ddots & \\ & & & I \end{bmatrix} - \lambda \begin{bmatrix} 0 & \cdots & 0 & -A_d \\ I & \ddots & & \\ & \ddots & 0 & \\ & & I & 0 \end{bmatrix}$$

is a strong linearization of $P(\lambda)$.

Proof. See Gohberg *et al.* [37, Proposition 1.1]. □

The connection between the eigenvalues and eigenvectors of the original matrix polynomial and the ones of the obtained linearization pencil can be summarized by the following theorem.

Theorem 2.5. *Let $P(\lambda)$ be defined by* (2.1) *and $\mathbf{L}(\lambda)$ by Theorem 2.4.*

1. *If the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$, then the pair $(\lambda_\star, \Lambda(\lambda_\star) \otimes x)$ with*

$$\Lambda(\lambda) := \begin{bmatrix} 1 \\ \lambda \\ \vdots \\ \lambda^{d-1} \end{bmatrix}$$

   is an eigenpair of $\mathbf{L}(\lambda)$.

2. *If the pair $(\lambda_\star, \mathbf{x})$ is an eigenpair of $\mathbf{L}(\lambda)$, then there exists a vector x such that $\mathbf{x} = \Lambda(\lambda_\star) \otimes x$ and the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$.*

Proof. From Theorem 2.4 we get the following identity

$$\mathbf{L}(\lambda)(\Lambda(\lambda) \otimes I) = e_1 \otimes P(\lambda). \tag{2.2}$$

Then, the proof of part *1* follows immediately from taking $\lambda = \lambda_\star$ and multiplying (2.2) from the right with x.

For the proof of part *2* we start with

$$\mathbf{L}(\lambda_\star)\mathbf{x} = (\mathbf{A} - \lambda_\star\mathbf{B})\mathbf{x} = 0.$$

Next, from the second till the last block row we find that

$$\begin{bmatrix} x^{[2]} \\ \vdots \\ x^{[d]} \end{bmatrix} = \begin{bmatrix} \lambda_\star \\ \vdots \\ \lambda_\star^{d-1} \end{bmatrix} \otimes x^{[1]},$$

and by choosing $x = x^{[1]}$ we obtain $\mathbf{x} = \Lambda(\lambda_\star) \otimes x$. Again evaluating (2.2) at $\lambda_\star$ and multiplying from the right with x complete the proof. □

The companion form linearizations for a matrix polynomial (2.1) can be generalized to vector spaces of pencils which serve as sources for other potential linearizations of $P(\lambda)$. For more details we refer to [69].

2.3 Lagrange basis

Although named after Lagrange [59], the original Lagrange form was firstly introduced by Waring in 1779 [118]. Rewriting it in its barycentric form results in a fast and stable way for polynomial interpolation.

Firstly, we discuss Lagrange interpolation in §2.3.1, followed by Hermite interpolation in §2.3.2. Thereafter, we discuss its linearizations for matrix polynomials in §2.3.3.

2.3.1 Lagrange interpolation

The original Lagrange form has certain shortcomings, e.g., increasing the degree of the polynomial by adding a new interpolation point requires computations from scratch and also the computation is numerically unstable [13]. Therefore, Lagrange interpolation is frequently considered as a bad choice for practical computations and thus mainly an analytic or theoretical tool for proving theorems. Nevertheless, rewriting in the so called modified Lagrange form and the barycentric Lagrange form overcomes the shortcomings of the original form and makes Lagrange interpolation very suitable for practical use.

Original Lagrange form

Suppose a function $f(\lambda)$ is sampled at $d+1$ distinct interpolation points (nodes) σ_i, $i = 0, \ldots, d$, with corresponding values $f_i := f(\sigma_i)$. The Lagrange interpolation problem addressed here is that of finding the polynomial $p(\lambda)$, of degree at most d, such that p interpolates f at the points σ_i, i.e.,

$$p(\sigma_i) = f_i, \qquad i = 0, \ldots, d.$$

This problem is well-posed and the solution can be written in *Lagrange form*:

$$p(\lambda) = \sum_{i=0}^{d} f_i \ell_i(\lambda), \tag{2.3}$$

where the *Lagrange polynomials* $\ell_i(\lambda)$ are defined as

$$\ell_i(\lambda) = \frac{\prod_{k=0,k\neq i}^{d}(\lambda - \sigma_k)}{\prod_{k=0,k\neq i}^{d}(\sigma_i - \sigma_k)}, \qquad i = 0, \ldots, d, \tag{2.4}$$

with the following property at the nodes

$$\ell_i(\sigma_k) = \begin{cases} 1, & i = k, \\ 0, & \text{otherwise}, \end{cases} \qquad i, k = 0, \ldots, d.$$

Modified Lagrange form

The original Lagrange formula (2.3) can be rewritten in such a way that it can be evaluated and updated in $O(d)$ operations [13]. Therefore, note that the numerator of $\ell_i(\lambda)$ in (2.4) can be written as the polynomial

$$\ell(\lambda) = (\lambda - \sigma_0)(\lambda - \sigma_1) \cdots (\lambda - \sigma_d) \tag{2.5}$$

divided by $\lambda - \sigma_i$. Defining the nonzero *barycentric weights* by

$$w_i = \frac{1}{\prod_{k \neq i}(\sigma_i - \sigma_k)}, \qquad i = 0, \ldots, d, \tag{2.6}$$

that is, $w_i = 1/\ell'(\sigma_i)$, allows us to write $\ell_i(\lambda)$ as

$$\ell_i(\lambda) = \ell(\lambda)\frac{w_i}{\lambda - \sigma_i}, \qquad i = 0, \ldots, d. \tag{2.7}$$

Now, note that all terms of the sum in (2.3) contain the factor $\ell(\lambda)$, which is independent of i. Bringing this factor in front of the sum yields the *modified Lagrange form* [13]:

$$p(\lambda) = \ell(\lambda) \sum_{i=0}^{d} f_i \frac{w_i}{\lambda - \sigma_i}. \tag{2.8}$$

This modified Lagrange form (2.8) is shown to be backward stable [46].

Barycentric Lagrange form

The modified Lagrange form (2.8) can still be modified to an even more elegant form. Therefore, we start from

$$1 = \sum_{i=0}^{d} \ell_i(\lambda) = \ell(\lambda) \sum_{i=0}^{d} \frac{w_i}{\lambda - \sigma_i}. \tag{2.9}$$

Dividing the modified Lagrange form for $p(\lambda)$ (2.8) by (2.9) and cancelling out the common factor $\ell(\lambda)$, we obtain the *barycentric form* [13]:

$$p(\lambda) = \frac{\displaystyle\sum_{i=0}^{d} f_i \frac{w_i}{\lambda - \sigma_i}}{\displaystyle\sum_{i=0}^{d} \frac{w_i}{\lambda - \sigma_i}} = \sum_{i=0}^{d} f_i b_i(\lambda), \tag{2.10}$$

where

$$b_i(\lambda) = \frac{1}{b(\lambda)} \cdot \frac{w_i}{\lambda - \sigma_i}, \qquad i = 0, \ldots, d,$$

with

$$b(\lambda) = \sum_{i=0}^{d} \frac{w_i}{\lambda - \sigma_i}.$$

The barycentric form is a Lagrange form, but one with a special symmetry. The weights w_i, still defined by (2.6), appear in the denominator exactly as in the numerator, except without the data factors f_i. Therefore, any common factor in all the weights w_i can be cancelled without affecting the value p.

Like the modified Lagrange form, the barycentric one also takes advantage of updating the weights w_i in $O(d)$ flops to incorporate a new data pair (σ_{d+1}, f_{d+1}). In [46] it is proved that the barycentric Lagrange interpolation form is forward stable for any set of interpolating points with a small Lebesgue constant. Finally, note that even if other weights w_i than (2.6) would be chosen in (2.10), the resulting rational function would still interpolate at the nodes σ_i in the sense that $p(\sigma_i) = f_i$.

2.3.2 Hermite interpolation

In the previous section, we considered interpolating polynomials in distinct points. Here we also allow multiple interpolation and review the Hermite interpolating Lagrange and barycentric polynomial.

Lagrange Hermite form

We still suppose that σ_i, $i = 0, \ldots, d$ are $d+1$ distinct interpolation points, but now with corresponding multiplicities m_i, with

$$m_0 + \cdots + m_n = D + 1,$$

where D is the degree of the corresponding interpolating polynomial $p(\lambda)$. The Lagrange form can now be generalized to multiple interpolation by

$$p(\lambda) = \sum_{i=0}^{d} \sum_{j=0}^{m_i - 1} \frac{f_i^{(j)}}{j!} \ell_{i,j}(\lambda),$$

where $f_i^{(j)} := f^{(j)}(\sigma_i)$ denotes the jth derivative of f evaluated at σ_i and

$$\ell_{i,j}(\lambda) = \ell(\lambda) \sum_{k=j}^{m_i - 1} \frac{w_{i,k}}{(\lambda - \sigma_i)^{k-j+1}},$$

is the generalization of (2.7) for multiple interpolation with

$$\ell(\lambda) = (\lambda - \sigma_0)^{m_0}(\lambda - \sigma_1)^{m_1} \cdots (\lambda - \sigma_n)^{m_n},$$

the generalization of (2.5). The constants $w_{i,j}$ are called the *generalized barycentric weights*. For the computation of these $w_{i,k}$ we refer to [23] and [87]. As in (2.8), we can bring the factor $\ell(\lambda)$ in front of the sums, yielding

$$p(\lambda) = \ell(\lambda) \sum_{i=0}^{d} \sum_{j=0}^{m_i-1} \frac{f_i^{(j)}}{j!} \sum_{k=j}^{m_i-1} \frac{w_{i,k}}{(\lambda - \sigma_i)^{k-j+1}}. \tag{2.11}$$

Barycentric Hermite form

The barycentric interpolating polynomial for multiple interpolation can be obtained in a similar way as in Section 2.3.1. Again, we start from

$$1 = \ell(\lambda) \sum_{i=0}^{d} \sum_{j=0}^{m_i-1} \frac{w_{i,j}}{(\lambda - \sigma_i)^{j+1}}. \tag{2.12}$$

Dividing the Lagrange Hermite form (2.11) by (2.12) and cancelling out the common factor $\ell(\lambda)$, we obtain the *barycentric Hermite form* [92]:

$$p(\lambda) = \frac{\displaystyle\sum_{i=0}^{d} \sum_{j=0}^{m_i-1} \frac{f_i^{(j)}}{j!} \sum_{k=j}^{m_i-1} \frac{w_{i,k}}{(\lambda - \sigma_i)^{k-j+1}}}{\displaystyle\sum_{i=0}^{d} \sum_{j=0}^{m_i-1} \frac{w_{i,j}}{(\lambda - \sigma_i)^{j+1}}} = \sum_{i=0}^{d} \sum_{j=0}^{m_i-1} \frac{f_i^{(j)}}{j!} b_{i,j}(\lambda),$$

where

$$b_{i,j}(\lambda) = \frac{1}{b(\lambda)} \sum_{k=j}^{m_i-1} \frac{w_{i,k}}{(\lambda - \sigma_i)^{k-j+1}}, \qquad i = 0, \ldots, d, \quad j = 0, \ldots, m_i - 1,$$

with

$$b(\lambda) = \sum_{i=0}^{d} \sum_{j=0}^{m_i-1} \frac{w_{i,j}}{(\lambda - \sigma_i)^{j+1}}.$$

2.3.3 Linearization

The first linearization of the Lagrange matrix polynomial

$$P(\lambda) = \ell(\lambda) \sum_{i=0}^{d} A_i \frac{w_i}{\lambda - \sigma_i}, \tag{2.13}$$

where $A_i \in \mathbb{C}^{n\times n}$ was introduced by Corless [24]. This linearization [24, 2, 3] can be extended to the barycentric Lagrange matrix polynomial

$$P(\lambda) = \frac{\displaystyle\sum_{i=0}^{d} A_i \frac{w_i}{\lambda - \sigma_i}}{\displaystyle\sum_{i=0}^{d} \frac{w_i}{\lambda - \sigma_i}} = \sum_{i=0}^{d} A_i b_i(\lambda), \tag{2.14}$$

by the following theorem.

Theorem 2.6. *Let $P(\lambda)$ be an $n \times n$ matrix polynomial of degree d in modified Lagrange form* (2.13) *or in barycentric Lagrange form* (2.14)*. Then the $(d+2)n \times (d+2)n$ linear companion pencil*

$$\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B},$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & A_0 & A_1 & \cdots & A_n \\ w_0 I & \sigma_0 I & & & \\ w_1 I & & \sigma_1 I & & \\ \vdots & & & \ddots & \\ w_n I & & & & \sigma_n I \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 & & & & \\ & I & & & \\ & & I & & \\ & & & \ddots & \\ & & & & I \end{bmatrix}, \tag{2.15}$$

is a linearization of $P(\lambda)$.

Remark that the arrowhead linearization of Theorem 2.6 is only a weak linearization since it introduces extra eigenvalues at infinity. In [3] it is proven that the pencil $\mathbf{A} - \lambda\mathbf{B}$, as defined in (2.15), is a strong linearization of

$$\widehat{P}(\lambda) := \lambda^{d+2}\, 0_n + \lambda^{d+1}\, 0_n + P(\lambda),$$

where $P(\lambda)$ is defined by (2.13) or (2.14). Therefore, a new linearization of dimension dn was proposed in [108] which yields a one-to-one mapping between the eigenstructure of the original matrix polynomial $P(\lambda)$ and the pencil $\mathbf{A}-\lambda\mathbf{B}$, corresponding to both finite eigenvalues and the eigenvalue at infinity.

We start by defining

$$\widetilde{\ell}_i(\lambda) := -\frac{\ell_i(\lambda)}{\lambda - \sigma_{i+1}} = -\ell(\lambda)\frac{w_i}{(\lambda - \sigma_i)(\lambda - \sigma_{i+1})}, \qquad i = 0, \ldots, d-1. \tag{2.16}$$

Next, using (2.16) for $i = d-1$ we can rewrite $\ell_d(\lambda)$ as follows

$$\ell_d(\lambda) = \frac{w_d}{w_{d-1}}(\sigma_{d-1} - \lambda)\,\widetilde{\ell}_{d-1}(\lambda). \tag{2.17}$$

Then, combining (2.16) and (2.17) yields

$$\begin{aligned}
P(\lambda) &= \sum_{i=0}^{d} A_i \ell_i(\lambda), \\
&= \sum_{i=0}^{d-1} A_i \ell_i(\lambda) + A_d \ell_d(\lambda), \\
&= \sum_{i=0}^{d-1} A_i (\sigma_{i+1} - \lambda)\, \widetilde{\ell}_i(\lambda) + A_d \frac{w_d}{w_{d-1}} (\sigma_{d-1} - \lambda)\, \widetilde{\ell}_{d-1}(\lambda), \\
&= \sum_{i=0}^{d-2} A_i (\sigma_{i+1} - \lambda)\, \widetilde{\ell}_i(\lambda) + \left[A_{d-1}(\sigma_d - \lambda) + A_d \frac{w_d}{w_{d-1}} (\sigma_{d-1} - \lambda) \right] \widetilde{\ell}_{d-1}(\lambda).
\end{aligned}$$

Similarly, for the barycentric Lagrange polynomial we define

$$\widetilde{b}_i(\lambda) := -\frac{b_i(\lambda)}{\lambda - \sigma_{i+1}} = -\frac{1}{b(\lambda)} \cdot \frac{w_i}{(\lambda - \sigma_i)(\lambda - \sigma_{i+1})}, \qquad i = 0, \ldots, d-1, \tag{2.18}$$

and using (2.18) for $i = d-1$ we rewrite $b_d(\lambda)$ as follows

$$b_d(\lambda) = \frac{w_d}{w_{d-1}} (\sigma_{d-1} - \lambda)\, \widetilde{b}_{d-1}(\lambda). \tag{2.19}$$

Combining (2.18) and (2.19) results in

$$P(\lambda) = \sum_{i=0}^{d-2} A_i (\sigma_{i+1} - \lambda)\, \widetilde{b}_i(\lambda) + \left[A_{d-1}(\sigma_d - \lambda) + A_d \frac{w_d}{w_{d-1}} (\sigma_{d-1} - \lambda) \right] \widetilde{b}_{d-1}(\lambda).$$

Before presenting the linearization, we formulate the relations between successive $\widetilde{\ell}_i(\lambda)$ and $\widetilde{b}_i(\lambda)$, respectively, in the following lemma.

Lemma 2.7. *Suppose $\widetilde{\ell}_i(\lambda)$ and $\widetilde{b}_i(\lambda)$ are defined by* (2.16) *and* (2.18)*, respectively. Let $\widetilde{p}_i(\lambda)$ be $\widetilde{\ell}_i(\lambda)$ or $\widetilde{b}_i(\lambda)$, then*

$$(\lambda - \sigma_{i-1})\, \widetilde{p}_{i-1}(\lambda) = \frac{w_{i-1}}{w_i} (\lambda - \sigma_{i+1})\, \widetilde{p}_i(\lambda),$$

for $i = 1, \ldots, d-1$.

Proof. The relations between $\widetilde{p}_{i-1}(\lambda)$ and $\widetilde{p}_i(\lambda)$ follow immediately from the definitions of $\widetilde{\ell}_i(\lambda)$ and $\widetilde{b}_i(\lambda)$. □

Theorem 2.8. *Let $P(\lambda)$ be an $n \times n$ matrix polynomial of degree d in modified Lagrange form* (2.13) *or in barycentric Lagrange form* (2.14)*. Then the $dn \times dn$ linear companion pencil*

$$\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B},$$

where

$$\mathbf{A} = \begin{bmatrix} \sigma_1 A_0 & \sigma_2 A_1 & \cdots & \sigma_{d-1} A_{d-2} & \sigma_d A_{d-1} + \sigma_{d-1}\theta_d^{-1} A_d \\ \sigma_0 I & -\sigma_2\theta_1 I & & & \\ & \ddots & \ddots & & \\ & & \ddots & -\sigma_{d-1}\theta_{d-2} I & \\ & & & \sigma_{d-2} I & -\sigma_d\theta_{d-1} I \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} A_0 & A_1 & \cdots & A_{d-2} & A_{d-1} + \theta_d^{-1} A_d \\ I & -\theta_1 I & & & \\ & \ddots & \ddots & & \\ & & \ddots & -\theta_{d-2} I & \\ & & & I & -\theta_{d-1} I \end{bmatrix},$$

with $\theta_i = w_{i-1}/w_i$ for $i = 1, \ldots, d$, is a strong linearization of $P(\lambda)$.

Proof. See Van Beeumen *et al.* [108, Theorem 4.8]. □

The connection between the eigenvalues and eigenvectors of the matrix polynomial in Lagrange form and the ones of the linearization pencil is summarized by the following theorem.

Theorem 2.9. *Let $P(\lambda)$ be defined by* (2.13) *or* (2.14) *and $\mathbf{L}(\lambda)$ by Theorem 2.8.*

1. *If the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$, then the pair $(\lambda_\star, \widetilde{\Lambda}(\lambda_\star) \otimes x)$ with*

$$\widetilde{\Lambda}(\lambda) := \begin{bmatrix} \widetilde{\ell}_0(\lambda) \\ \widetilde{\ell}_1(\lambda) \\ \vdots \\ \widetilde{\ell}_{n-1}(\lambda) \end{bmatrix}$$

   is an eigenpair of $\mathbf{L}(\lambda)$.

2. *If the pair $(\lambda_\star, \mathbf{x})$ is an eigenpair of $\mathbf{L}(\lambda)$, then there exists a vector x such that $\mathbf{x} = \widetilde{\Lambda}(\lambda_\star) \otimes x$ and the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$.*

Proof. From Theorem 2.8 we get the following identity

$$\mathbf{L}(\lambda)(\widetilde{\Lambda}(\lambda) \otimes I) = e_1 \otimes P(\lambda). \tag{2.20}$$

Then, the proof of part *1* follows from taking $\lambda = \lambda_\star$ and multiplying (2.20) from the right with x. Furthermore, since $\widetilde{\Lambda}(\lambda) \neq 0$ for all λ, we have $\widetilde{\Lambda}(\lambda_\star) \otimes x \neq 0$.

For the proof of part *2* we start with

$$\mathbf{L}(\lambda_\star)\mathbf{x} = (\mathbf{A} - \lambda_\star \mathbf{B})\mathbf{x} = 0.$$

Next, from the second till the last block row we find that

$$(\lambda - \sigma_{i-1})\, x^{[i]} = \frac{w_{i-1}}{w_i}(\lambda - \sigma_{i+1})\, x^{[i+1]}, \qquad i = 1, \ldots, d-1.$$

By using the relations of Lemma 2.7 and by choosing $x = x^{[i]}/\widetilde{\ell}_i(\lambda_\star)$, with $\widetilde{\ell}_i(\lambda_\star) \neq 0$, we obtain $\mathbf{x} = \widetilde{\Lambda}(\lambda_\star) \otimes x$. Again evaluating (2.20) at $\lambda_\star$ and multiplying from the right with x complete the proof. □

The linearizations of Theorem 2.6 and Theorem 2.8 can also be generalized to Hermite interpolation, see [93, 25] and [108], respectively.

2.4 Orthogonal bases

In contrast to the Lagrange polynomials, orthogonal polynomials form degree-graded bases. Firstly, we discuss the three-term recurrence relation in §2.4.1. Next, we describe a general linearization for all orthogonal bases in §2.4.2.

2.4.1 Recurrence relation

In general, we call any sequence of polynomials $p_0(\lambda), p_1(\lambda), \ldots$ with $p_i(\lambda)$ of degree i a degree-graded basis. Consequently, $p_i(\lambda)$ can be written as a linear combination of $p_0(\lambda), p_1(\lambda), \ldots, p_{i-1}(\lambda)$. Furthermore, if these polynomials are orthonormal on an interval of the real line, then they satisfy the following three-term recurrence relation [27]

$$\lambda p_i(\lambda) = \alpha_i p_{i+1}(\lambda) + \beta_i p_i(\lambda) + \gamma_i p_{i-1}(\lambda), \qquad i = 0, 1, \ldots, \tag{2.21}$$

where $p_{-1}(\lambda) \equiv 0$, $p_0(\lambda) \equiv 1$, $\alpha_i \neq 0$, and $\gamma_i > 0$. The specific values of α_i, β_i, and γ_i for some orthogonal bases are summarized in Table 2.1. Remark that the monomials also satisfy a recurrence relation (2.21) with $\alpha_i = 1$ and $\beta_i = \gamma_i = 0$.

Table 2.1: Orthogonal bases with their three-term recurrence coefficients.

Basis	α_i	β_i	γ_i
Chebyshev (first and second kind)	$\frac{1}{2}$	0	$\frac{1}{2}$
Hermite	$\frac{1}{2}$	0	i
Laguerre	$-(i+1)$	$2i+1$	$-i$
Legendre	$\frac{i+1}{2i+1}$	0	$\frac{i}{2i+1}$

2.4.2 Linearization

Let a degree-graded matrix polynomial be defined as follows

$$P(\lambda) = \sum_{i=0}^{d} A_i p_i(\lambda), \tag{2.22}$$

where $A_i \in \mathbb{C}^{n\times n}$ and $p_i(\lambda)$ satisfying a three-term recurrence relation (2.21). Then, the corresponding linearization and the eigenpair connection are given by the following theorems.

Theorem 2.10. *Let $P(\lambda)$ be an $n\times n$ degree-graded matrix polynomial of degree d in the form* (2.22)*. Then the $dn \times dn$ linear companion pencil*

$$\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B},$$

where

$$\mathbf{A} = \begin{bmatrix} A_0 & A_1 & A_2 & \dots & A_{d-3} & A_{d-2} - \frac{\gamma_{d-1}}{\alpha_{d-1}}A_d & A_{d-1} - \frac{\beta_{d-1}}{\alpha_{d-1}}A_d \\ \beta_0 I & \alpha_0 I & & & & & \\ \gamma_1 I & \beta_1 I & \alpha_1 I & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \alpha_{d-4} I & & \\ & & & \ddots & \beta_{d-3} I & \alpha_{d-3} I & \\ & & & & \gamma_{d-2} I & \beta_{d-2} I & \alpha_{d-2} I \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -\frac{1}{\alpha_{d-1}}A_d \\ I & 0 & & & \\ & \ddots & \ddots & & \\ & & \ddots & 0 & \\ & & & I & 0 \end{bmatrix},$$

is a strong linearization of $P(\lambda)$.

Proof. See Amiraslani *et al.* [3, Theorem 3.1]. □

Note that for general degree-graded bases, i.e., satisfying an i-term recurrence relation, the lower triangular part of the matrices $\mathbf{A}$ and $\mathbf{B}$ in Theorem 2.10 can completely fill up.

Theorem 2.11. *Let $P(\lambda)$ be defined by (2.22) and $\mathbf{L}(\lambda)$ by Theorem 2.10.*

1. *If the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$, then the pair $(\lambda_\star, p(\lambda_\star) \otimes x)$ with*
$$p(\lambda) := \begin{bmatrix} p_0(\lambda) \\ p_1(\lambda) \\ \vdots \\ p_{d-1}(\lambda) \end{bmatrix}$$
is an eigenpair of $\mathbf{L}(\lambda)$.

2. *If the pair $(\lambda_\star, \mathbf{x})$ is an eigenpair of $\mathbf{L}(\lambda)$, then there exists a vector x such that $\mathbf{x} = p(\lambda_\star) \otimes x$ and the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$.*

Proof. From Theorem 2.10 we get the following identity

$$\mathbf{L}(\lambda)(p(\lambda) \otimes I) = e_1 \otimes P(\lambda). \tag{2.23}$$

Then, the proof of part *1* immediately follows from taking $\lambda = \lambda_\star$ and multiplying (2.23) from the right with x. Furthermore, by definition $p_0(\lambda) \equiv 1$. Hence, we have $p(\lambda_\star) \otimes x \neq 0$.

For the proof of part *2* we start with

$$\mathbf{L}(\lambda_\star)\mathbf{x} = (\mathbf{A} - \lambda_\star \mathbf{B})\mathbf{x} = 0.$$

Next, from the second till the last block row we find that

$$\begin{aligned} \beta_0 x^{[1]} + \alpha_0 x^{[2]} &= \lambda x^{[1]}, \\ \gamma_{i-1} x^{[i-1]} + \beta_{i-1} x^{[i]} + \alpha_{i-1} x^{[i+1]} &= \lambda x^{[i]}, \qquad i = 2, \ldots, d-1. \end{aligned}$$

By using the three-term recurrence relation (2.21) and choosing $x = x^{[1]}$, we obtain $\mathbf{x} = p(\lambda_\star) \otimes x$. Again evaluating (2.23) at $\lambda_\star$ and multiplying from the right with x complete the proof. □

Similarly as for the monomial basis, the companion form linearizations of orthogonal polynomials can be generalized to vector spaces. For more details we refer to [101].

2.5 Newton basis

As discussed in Section 2.3, an interpolating polynomial in Lagrange form can easily be constructed as a linear combination of the function values at the interpolation points. Furthermore, incorporating a new data pair can efficiently be done and only involves scalar operations. However, incorporating a new data pair changes all the basis functions and will also affect the linearization pencils.

On the other hand, increasing the degree of an interpolating polynomial in Newton form, by adding a new data pair, will leave both the basis functions as well as the divided differences unchanged. Consequently, the linearization matrices of lower degree interpolating polynomials form submatrices of the higher degree ones.

Before presenting the linearization of Newton matrix polynomials in §2.5.2, we review Newton interpolation in §2.5.1.

2.5.1 Newton interpolation

The Newton polynomial which interpolates the function $f(\lambda)$ in distinct nodes $\sigma_0, \sigma_1, \ldots, \sigma_d$ is defined as follows

$$p(\lambda) = \sum_{i=0}^{d} \alpha_i n_i(\lambda), \tag{2.24}$$

where the Newton basis functions are

$$\begin{aligned} n_0(\lambda) &:= 1, \\ n_i(\lambda) &:= \prod_{k=1}^{i} (\lambda - \sigma_{k-1}), \qquad i = 1, 2, \ldots, \end{aligned}$$

with the recurrence relation

$$n_i(\lambda) = (\lambda - \sigma_{i-1})\, n_{i-1}(\lambda), \qquad i = 1, 2, \ldots.$$

The coefficients α_i are the *divided differences*

$$\begin{aligned} \alpha_0 := f[\sigma_0] &= f(\sigma_0), \\ \alpha_i := f[\sigma_0, \ldots, \sigma_i] &= \frac{f[\sigma_1, \ldots, \sigma_i] - f[\sigma_0, \ldots, \sigma_{i-1}]}{\sigma_i - \sigma_0}, \end{aligned}$$

which can be computed from a divided differences table. However, this way is often numerically very unstable. Therefore, we use the following connection between divided differences and matrix functions [79, 28, 47].

Theorem 2.12 (Opitz [79]). *The divided difference $\alpha_0, \ldots, \alpha_k$ are the elements in the first row of $f(M)$, where*

$$M = \begin{bmatrix} \sigma_0 & 1 & & & \\ & \sigma_1 & 1 & & \\ & & \ddots & \ddots & \\ & & & \ddots & 1 \\ & & & & \sigma_k \end{bmatrix}.$$

The interpolating Newton polynomial in Hermite form has the same form as (2.24), but allows interpolation nodes with multiplicities higher than one. Hence, the computation of the divided differences involves not only the function value but also higher order derivatives.

2.5.2 Linearization

Let the Newton matrix polynomial be defined as follows

$$P(\lambda) = \sum_{i=0}^{d} A_i n_i(\lambda), \tag{2.25}$$

where $A_i \in \mathbb{C}^{n\times n}$ are the divided difference matrices. Then, the corresponding linearization and the eigenpair connection are given by the following theorems.

Theorem 2.13. *Let $P(\lambda)$ be an $n \times n$ matrix polynomial of degree d in Newton from* (2.25)*. Then the $dn \times dn$ linear companion pencil*

$$\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B},$$

where

$$\mathbf{A} = \begin{bmatrix} A_0 & A_1 & \ldots & A_{d-2} & A_{d-1} - \sigma_{d-1}A_d \\ \sigma_0 I & I & & & \\ & \ddots & \ddots & & \\ & & \ddots & I & \\ & & & \sigma_{d-2}I & I \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -A_d \\ I & 0 & & & \\ & \ddots & \ddots & & \\ & & \ddots & 0 & \\ & & & I & 0 \end{bmatrix},$$

is a strong linearization of $P(\lambda)$.

Proof. See Amiraslani *et al.* [3, Theorem 3.1]. □

Although the Newton basis is not orthogonal, the basis functions also satisfy a three-term recurrence relation (2.21). Therefore, Theorem 2.13 can be seen as a special case of Theorem 2.10 with $\alpha_i = 1$, $\beta_i = \sigma_i$, and $\gamma_i = 0$.

Theorem 2.14. *Let $P(\lambda)$ be defined by* (2.25) *and $\mathbf{L}(\lambda)$ by Theorem 2.13.*

1. *If the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$, then the pair $(\lambda_\star, n(\lambda_\star) \otimes x)$ with*
$$n(\lambda) := \begin{bmatrix} n_0(\lambda) \\ n_1(\lambda) \\ \vdots \\ n_{d-1}(\lambda) \end{bmatrix}$$
is an eigenpair of $\mathbf{L}(\lambda)$.

2. *If the pair $(\lambda_\star, \mathbf{x})$ is an eigenpair of $\mathbf{L}(\lambda)$, then there exists a vector x such that $\mathbf{x} = n(\lambda_\star) \otimes x$ and the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$.*

Proof. See proof of Theorem 2.11 with $\alpha_i = 1$, $\beta_i = \sigma_i$, and $\gamma_i = 0$. □

Chapter 3

Dynamic Polynomial Interpolation

We consider the nonlinear eigenvalue problem (NLEP)

$$A(\lambda)x = 0, \tag{3.1}$$

where $\lambda \in \Omega \subseteq \mathbb{C}$ and $x \in \mathbb{C}^n \setminus \{0\}$. We also assume that the matrix-valued function $A : \Omega \to \mathbb{C}^{n\times n}$ is analytic on Ω. Note that the NLEP (3.1) can have infinitely many solutions and the eigenvectors can be linearly dependent. Therefore, we will focus on finding eigenvalues in a specific target set $\Sigma \subset \Omega$.

In order to compute the eigenvalues of (3.1) lying in Σ, we first approximate the matrix-valued function $A(\lambda)$ by an interpolating matrix polynomial. Next, a companion-type linearization is applied to obtain a larger but linear standard eigenvalue problem. Furthermore, by making use of matrix polynomials expressed in the monomial basis or Newton basis together with solving the generalized eigenvalue problem via (rational) Krylov methods with appropriate shift(s), we obtain dynamic algorithms which rely on dynamic polynomial interpolation where neither the degree of the interpolating polynomial, nor the interpolation nodes needed to be fixed in advance.

This chapter is organized as follows. Firstly, we introduce the main idea of dynamic polynomial interpolation in the monomial basis with the Taylor–Arnoldi method in Section 3.1. In this method there is a dynamic interplay of interpolation, linearization, and Arnoldi's method. Next, this idea is applied to Newton interpolation in Section 3.2, resulting in the Newton Rational Krylov method. In Section 3.3 the methods are illustrated with some numerical examples. Finally, we summarise the main results in Section 3.4.

3.1 Taylor–Arnoldi method

The Taylor–Arnoldi method for solving nonlinear eigenvalue problems (3.1) was firstly introduced by Jarlebring *et al.* [50] and relies on the following 3 main ingredients:

1. the approximation of the matrix-valued function $A(\lambda)$ in (3.1) by truncated Taylor series of increasing degree;
2. the property that increasing the degree of a matrix polynomial, expressed in the monomial basis, corresponds to only adding one block row and column to the linearization pencil, while the other part remains unchanged;
3. solving the generalized linear eigenvalue problem, obtained after linearization, by Arnoldi's method with shift equal to the center point of the Taylor series and a particular starting vector.

The interweaving of the Taylor approximation with the Arnoldi method yields a linearization pencil that grows in every iteration. However, its structure can be exploited such that the method only involves linear algebra operations with matrices of the original NLEP dimension n.

Firstly, we introduce the basic idea in §3.1.1 and §3.1.2 with Maclaurin series, which simplifies the notation. Secondly, we generalize to Taylor series and introduce the Taylor–Arnoldi algorithm in §3.1.3.

3.1.1 A companion-type reformulation

Consider the truncated Maclaurin expansion of $A(\lambda)$ in (3.1)

$$A(\lambda) \approx P(\lambda) = \sum_{i=0}^{d} \frac{A^{(i)}(0)}{i!}\lambda^i, \tag{3.2}$$

where $i!$ denotes the factorial of i and $A^{(i)}(0)$ the ith derivative of the matrix-valued function $A(\lambda)$ evaluated at zero. Then, the NLEP (3.1) can be approximated by the following PEP

$$P(\lambda)x = 0,$$

where $P(\lambda)$ is defined by (3.2). Next, by using Theorem 2.4, this PEP can be linearized as follows

$$\underbrace{\begin{bmatrix} A_0 & A_1 & \cdots & A_{d-1} \\ & I & & \\ & & \ddots & \\ & & & I \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} x \\ \lambda x \\ \vdots \\ \lambda^{d-1}x \end{bmatrix}}_{\mathbf{x}} = \lambda \underbrace{\begin{bmatrix} 0 & \cdots & 0 & -A_d \\ I & \ddots & & \\ & \ddots & 0 & \\ & & I & 0 \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} x \\ \lambda x \\ \vdots \\ \lambda^{d-1}x \end{bmatrix}}_{\mathbf{x}}, \tag{3.3}$$

where $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{dn \times dn}$, $\mathbf{x} \in \mathbb{C}^{dn}$, and $A_i = A^{(i)}(0)/(i!)$ for $i = 0, 1, \ldots, d$.

3.1.2 Building the Krylov subspace

The idea now is to solve the generalized linear eigenvalue problem (3.3) with the Arnoldi method in order to compute approximations for the eigenvalues of the original nonlinear eigenvalue problem (3.1). We start with the following lemma.

Lemma 3.1. *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be defined as in* (3.3)*. Then,*

$$\mathbf{S}_d := \mathbf{A}^{-1}\mathbf{B} = \begin{bmatrix} S_1 & S_2 & \cdots & S_d \\ I & 0 & & \\ & \ddots & \ddots & \\ & & I & 0 \end{bmatrix},$$

where $S_i = -A_0^{-1}A_i$ *for* $i = 1, \ldots, d$.

Proof. The proof immediately follows from the definition of $\mathbf{A}$ and $\mathbf{B}$. □

Using Lemma 3.1, the shift and invert step in every iteration j of Algorithm 1.2 with shift $\sigma = 0$ yields the following matrix-vector multiplication

$$\widehat{\mathbf{v}} = \mathbf{A}^{-1}\mathbf{B}\mathbf{v}_j = \mathbf{S}_d\mathbf{v}_j. \tag{3.4}$$

Furthermore, suppose that only the first block component of the starting vector $\mathbf{v}_1 \in \mathbb{C}^{dn}$ is nonzero

$$\mathbf{v}_1 = \begin{bmatrix} v_1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}, \tag{3.5}$$

with $v_1 \in \mathbb{C}^n$. Then, by using (3.4), the Krylov subspace is spanned by the following structured vectors

$$\mathcal{K}(\mathbf{A}, \mathbf{B}, \mathbf{v}_1) = \operatorname{span}\left\{ \begin{bmatrix} v_1 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix}, \begin{bmatrix} v_2 = S_1 v_1 \\ v_1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}, \begin{bmatrix} v_3 = S_1 v_2 + S_2 v_1 \\ v_2 \\ v_1 \\ 0 \\ \vdots \end{bmatrix}, \cdots \right\}.$$

Consequently, the Krylov subspace $\mathbf{V}$ will have the following block structure

$$\mathbf{V} = \begin{bmatrix} \star & \star & \star & \star \\ & \star & \star & \star \\ & & \star & \star \\ & & & \star \\ & & & \\ & & & \end{bmatrix}.$$

The advantages of choosing (3.5) as a starting vector for the Arnoldi method are summarised in the following lemma and proposition.

Lemma 3.2. *Suppose that the starting vector of the Arnoldi method is of the form* (3.5)*. Then for all j, $1 \leq j \leq d-1$, only the first j blocks of the vector $\mathbf{v}_j$ are nonzero.*

Proof. We prove this lemma by induction and start with $j = 1$. By definition (3.5) only the first block of $\mathbf{v}_1$ is nonzero. Now suppose that the lemma holds for j; then only the first j blocks of $\mathbf{v}_j$ are nonzero. By applying Lemma 3.1, we find that only the first $j+1$ blocks of $\widehat{\mathbf{v}} = \mathbf{A}^{-1}\mathbf{B}\mathbf{v}_j = \mathbf{S}_d\mathbf{v}_j$ are nonzero and thus so do the first $j+1$ ones of $\mathbf{v}_{j+1}$. This completes the proof. □

An important consequence of Lemma 3.2 is that, at each iteration j of the Arnoldi method, we only use $S_1, \ldots, S_j$ in order to compute the next vector $\mathbf{v}_{j+1}$. Consequently, only the first $j+1$ Taylor coefficients are used.

Proposition 3.3. *The Ritz values λ_i, computed at iteration j of the Arnoldi method are independent of d as long as $j < d$.*

Proof. At iteration j, the Ritz values are computed from the upper $j \times j$ parts of the Hessenberg matrix $\underline{H}_j$, which is obtained from the orthogonalization process of only $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{j+1}$. Following Lemma 3.2, only the first $j+1$ Taylor coefficients $A_0, A_1, \ldots, A_j$ are used for the construction of the subspace $\mathbf{V}_{j+1}$. Therefore, the approximated eigenvalues are independent of $A_{j+1}, \ldots, A_d$. Hence they are also independent of d, which proves the proposition. □

Remark 3.4. Performing j steps of the Taylor–Arnoldi method, with an appropriate starting vector, produces the same subspace as j steps of the standard Arnoldi method with the matrices $\mathbf{A}$ and $\mathbf{B}$ for any $d > j$. Taking a limit argument, $d \to \infty$, the Taylor–Arnoldi method can be interpreted as an Arnoldi method directly applied to an infinite dimensional linear problem equivalent to the original nonlinear problem [52]. However, only finite arithmetic is used, i.e., standard linear algebra operations applied to matrices of finite size. This connection is fully elaborated in Chapter 5.

3.1.3 Algorithm

We first generalize the Maclaurin expansion to a Taylor expansion at an arbitrary point σ in the complex plane. Therefore, we approximate $A(\lambda)$ in (3.1) by the following truncated Taylor expansion

$$A(\lambda) \approx P(\lambda) = \sum_{i=0}^{d} \frac{A^{(i)}(\sigma)}{i!}(\lambda-\sigma)^i =: \sum_{i=0}^{d} A_i \frac{(\lambda-\sigma)^i}{\alpha_i}, \tag{3.6}$$

where $\alpha_0, \alpha_1, \ldots, \alpha_d$ are nonzero scaling parameters and $A_i = \alpha_i A^{(i)}(\sigma)/(i!)$ for $i = 0, 1, \ldots, d$. Next, the NLEP (3.1) can be approximated by the following GEP

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{B}\mathbf{x},$$

where $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{dn \times dn}$

$$\mathbf{A} = \begin{bmatrix} A_0 & A_1 & \cdots & A_{d-2} & A_{d-1} - \sigma A_d/\beta_d \\ \sigma I & \beta_1 I & & & \\ & \ddots & \ddots & & \\ & & \ddots & \beta_{d-2} I & \\ & & & \sigma I & \beta_{d-1} I \end{bmatrix}, \tag{3.7}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -A_d/\beta_d \\ I & 0 & & & \\ & \ddots & \ddots & & \\ & & \ddots & 0 & \\ & & & I & 0 \end{bmatrix}, \tag{3.8}$$

with $\beta_i = \alpha_i/\alpha_{i-1}$ for $i = 1, \ldots, d$, and $\mathbf{x} \in \mathbb{C}^{dn}$

$$\mathbf{x} = \begin{bmatrix} 1/\alpha_0 \cdot x \\ (\lambda-\sigma)/\alpha_1 \cdot x \\ \vdots \\ (\lambda-\sigma)^{d-1}/\alpha_{d-1} \cdot x \end{bmatrix}.$$

Note that, by choosing $\sigma = 0$ and $\alpha_i = \beta_i = 1$, we obtain exactly the same linearization as described in §3.1.1. Let $\mathbf{A}$ and $\mathbf{B}$ be defined by (3.7) and (3.8), respectively, with $\beta_i = 1$. Then, $\mathbf{S}_d := (\mathbf{A} - \sigma\mathbf{B})^{-1}\mathbf{B}$ is exactly the same matrix as in Lemma 3.1. Consequently, Lemma 3.2 and Proposition 3.3 remain valid.

The scaling parameters α_i in (3.6) are chosen in [52] as $\alpha_i = i!$, yielding $A_i = A^{(i)}(\sigma)$. For a more detailed study of the basis functions scaling, we refer to the next chapter.

Based on Lemmas 3.1–3.2 and Proposition 3.3, the Taylor–Arnoldi algorithm for solving the NLEP (3.1) can be dynamically implemented. Algorithm 3.1 gives an outline and Figure 3.1 shows a visualization. We can subdivide each iteration of Algorithm 3.1 into two main phases: an expansion phase followed by an Arnoldi phase.

Algorithm 3.1: Taylor–Arnoldi method [50]

1 Choose shift σ and starting vector $v_1 \in \mathbb{C}^n$.
for $j = 1, 2, \ldots$ **do**
  EXPANSION PHASE:
2 Compute next Taylor coefficient: $A_j = \frac{A^{(j)}(\sigma)}{j!}$.
3 Expand $\mathbf{S}_j$ and $\mathbf{V}_j$.
  ARNOLDI PHASE:
4 Compute $\widehat{\mathbf{v}} := \mathbf{S}_j \mathbf{v}_j$.
5 Orthogonalize: $\widetilde{\mathbf{v}} := \widehat{\mathbf{v}} - \mathbf{V}_j h_j$, where $h_j = \mathbf{V}_j^* \widehat{\mathbf{v}}$.
6 Get new vector: $\mathbf{v}_{j+1} = \widetilde{\mathbf{v}}/h_{j+1,j}$, where $h_{j+1,j} = \|\widetilde{\mathbf{v}}\|$.
7 Compute Ritzpairs: $(\lambda_i, \mathbf{x}_i)$.
8 Nonlinear eigenpairs: $\left(\lambda_i, x_i^{[1]}\right)$ and test for convergence.
end

Firstly, in the expansion phase (lines 2–3), we compute at iteration j the next Taylor coefficient A_j in order to extend the matrix $\mathbf{S}_{j-1}$ into $\mathbf{S}_j$. We also extend the matrix $\mathbf{V}_j$ with a zero block at the bottom.

Secondly, in the Arnoldi phase (lines 4–7), we perform an Arnoldi step with the extended matrix $\mathbf{S}_j$. In line 7, the Ritz values λ_i are computed as follows

$$\lambda_i = \sigma + \frac{1}{\theta_i}, \qquad i = 1, \ldots, j,$$

with θ_i the eigenvalues of the low-dimensional linear eigenvalue problem

$$H_j s_i = \theta_i s_i, \qquad s_i \neq 0,$$

where H_j is the upper $j \times j$ part of the Hessenberg matrix $\underline{H}_j$, obtained from the orthogonalization process. The Ritz vectors $\mathbf{x}_i$ are obtained by left multiplication of s_i with $\mathbf{V}_{j+1}\underline{H}_j$.

Finally, in line 8, we take the first blocks of $\mathbf{x}_i$ as approximations for the nonlinear eigenvectors and check for convergence of the nonlinear eigenpairs $\left(\lambda_i, x_i^{[1]}\right)$.

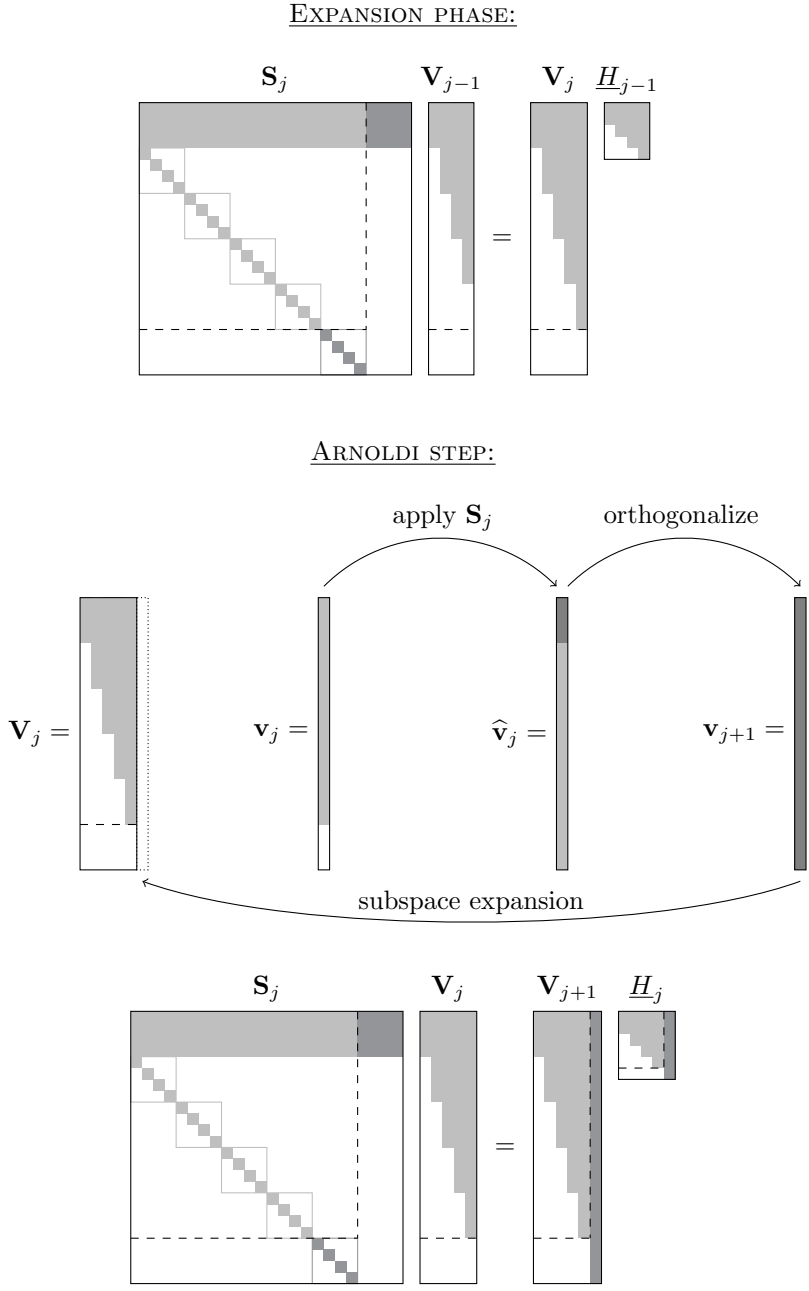


Figure 3.1: Visualization of Algorithm 3.1 [52].

3.2 Newton Rational Krylov method

The Newton Rational Krylov method [106] for solving nonlinear eigenvalue problems (3.1) generalizes the dynamic polynomial approximation properties of the Taylor–Arnoldi method to Newton/Hermite interpolation. Its 3 main ingredients are:

1. the approximation of the matrix-valued function $A(\lambda)$ in (3.1) by truncated interpolating matrix polynomials in Newton/Hermite form of increasing degree;
2. the property that increasing the degree of a matrix polynomial, expressed in Newton basis, corresponds to only adding one block row and column to the linearization pencil, while the other part remains unchanged;
3. solving the generalized linear eigenvalue problem, obtained after linearization, by a rational Krylov method with a particular starting vector and where the shifts are no free parameters but equal to the interpolation points for approximating $A(\lambda)$.

By using a (Hermite) interpolating polynomial to approximate $A(\lambda)$, the expectations are that a better approximation can be obtained than a truncated Taylor expansion of the same degree. The advantage of using polynomial interpolation in Newton form is that adding a new interpolation point just adds a new polynomial to the basis. This is in contrast to for example polynomial interpolation in Lagrange form where an extra interpolation point changes all the basis functions. Therefore, Newton/Hermite interpolation allows for iteratively adding new points in a flexible way, which implies that the linearization grows in every iteration. When, in addition, we choose the shifts of the rational Krylov method equal to the interpolation points, the rational Krylov expansion on the linearized problem takes advantage of the specific structure, so that the rational Krylov method can be interpreted as a rational Krylov method applied to a fixed size matrix (that does not grow during the iterations). This property makes the process dynamic and has the important consequence that the interpolation points need not to be fixed in advance. In each iteration we can choose a new interpolation point based on the results of the previous ones. As in the Taylor–Arnoldi method, the companion structure of the linearization pencil can be exploited such that only linear algebra operations with matrices of the original NLEP dimension n are needed. For a detailed analysis of maximally exploiting the pencil structure we refer to Chapter 6.

Firstly, we discuss in §3.2.1 and §3.2.2 the idea of interweaving Newton/Hermite interpolation with the rational Krylov method. Secondly, we introduce the Newton Rational Krylov algorithm in §3.2.3. Thirdly, we discuss low rank exploitation of the coefficient matrices in §3.2.4. Finally, we describe briefly its connection with Newton's method in §3.2.5.

3.2.1 A companion-type reformulation

Let us start with the matrix polynomial $P(\lambda)$ which (Hermite) interpolates the matrix-valued function $A(\lambda)$ at the interpolation points $\sigma_0, \sigma_1, \ldots, \sigma_d$

$$A(\lambda) \approx P(\lambda) := \sum_{i=0}^{d} A_i n_i(\lambda), \tag{3.9}$$

where $A_i \in \mathbb{C}^{n\times n}$ are the divided difference matrices and

$$n_i(\lambda) := \frac{1}{\beta_0} \prod_{k=1}^{i} \frac{1}{\beta_k}(\lambda - \sigma_{k-1}), \qquad i = 0, 1, \ldots, d, \tag{3.10}$$

the scaled Newton basis functions with $\beta_0, \beta_1, \ldots, \beta_d$ nonzero scaling parameters. Then, the NLEP (3.1) can be approximated by the following PEP

$$P(\lambda)x = 0,$$

where $P(\lambda)$ is defined by (3.9). Next, similarly to Theorem 2.13, this PEP can be linearized as follows

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{B}\mathbf{x}, \tag{3.11}$$

where

$$\mathbf{A} = \begin{bmatrix} A_0 & A_1 & \cdots & A_{d-2} & A_{d-1} - \sigma_{d-1}A_d/\beta_d \\ \sigma_0 I & \beta_1 I & & & \\ & \ddots & \ddots & & \\ & & \ddots & \beta_{d-2} I & \\ & & & \sigma_{d-2} I & \beta_{d-1} I \end{bmatrix}, \tag{3.12}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -A_d/\beta_d \\ I & 0 & & & \\ & \ddots & \ddots & & \\ & & \ddots & 0 & \\ & & & I & 0 \end{bmatrix}, \tag{3.13}$$

and

$$\mathbf{x} = \begin{bmatrix} n_0(\lambda)x \\ n_1(\lambda)x \\ \vdots \\ n_{d-1}(\lambda)x \end{bmatrix}.$$

Note that by choosing all scaling parameters $\beta_i = 1$, the matrices (3.12)–(3.13) are exactly the same as in Theorem 2.13.

3.2.2 Building the rational Krylov subspace

As in the Taylor–Arnoldi method, we solve the generalized eigenvalue problem (3.11) in order to compute approximations for the eigenvalues of the original nonlinear eigenvalue problem (3.1). However, we now need a rational Krylov method for applying the "trick" with the dynamically growing linearization. We start with the following key lemma.

Lemma 3.5. *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be defined by* (3.12)–(3.13)*. Suppose that only the first* $j+1$ *blocks of the vector* $\mathbf{y} \in \mathbb{C}^{dn}$ *are nonzero. Then for all* j*,* $0 \leq j \leq d-2$*, only the first* $j+1$ *blocks of the solution* $\mathbf{x}$ *of the linear system with shift* σ_j

$$(\mathbf{A} - \sigma_j \mathbf{B})\mathbf{x} = \mathbf{y}, \tag{3.14}$$

are nonzero. Furthermore, only the leading submatrices

$$\mathbf{A}_j = \begin{bmatrix} A_0 & A_1 & \dots & A_j \\ \sigma_0 I & \beta_1 I & & \\ & \ddots & \ddots & \\ & & \sigma_{j-1} I & \beta_j I \end{bmatrix}, \qquad \mathbf{B}_j = \begin{bmatrix} 0 & & & \\ I & 0 & & \\ & \ddots & \ddots & \\ & & I & 0 \end{bmatrix},$$

of $\mathbf{A}$ *and* $\mathbf{B}$*, respectively, are needed to compute this nonzero part of* $\mathbf{x}$*.*

Proof. This lemma directly follows from the fact that the matrix $\mathbf{A} - \sigma_j \mathbf{B}$ in (3.14) is block triangular. □

The main difference to a standard rational Krylov method [83, 85] is that the shifts in the Newton Rational Krylov method are not free parameters, but they are implicitly prescribed by the matrices $\mathbf{A}$ and $\mathbf{B}$ in (3.12)–(3.13), namely the nodes $\sigma_1, \sigma_2, \ldots$. Similarly as in §3.1.2, we suppose that only the first block component of the starting vector $\mathbf{v}_1 \in \mathbb{C}^{dn}$ is nonzero

$$\mathbf{v}_1 = \begin{bmatrix} v_1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}, \tag{3.15}$$

with $v_1 \in \mathbb{C}^n$. Consequently by Lemma 3.5, the rational Krylov subspace $\mathbf{V}$ has now also a growing block structure. This is summarized in the following lemma.

Lemma 3.6. *Suppose that the starting vector* $\mathbf{v}_1$ *of the rational Krylov method is of the form* (3.15) *and we use the nodes* $\sigma_1, \sigma_2, \ldots$ *as shifts. Then for all* j*,* $1 \leq j \leq d-1$*, only the first* j *blocks of the vector* $\mathbf{v}_j$*, generated by the rational Krylov method, are nonzero.*

Proof. We prove this lemma by induction and start with $j = 1$. By definition (3.15) only the first block of $\mathbf{v}_1$ is nonzero. Now suppose that the lemma holds for j; then only the first j blocks of $\mathbf{v}_j$ are nonzero and so do the first blocks of the continuation vector $\mathbf{w}_j$. Applying $\mathbf{B}$ to $\mathbf{w}_j$ results in a down-shift of the nonzero part of $\mathbf{w}_j$ by one block. Then by Lemma 3.5, we find that only the first $j+1$ blocks of $\widetilde{\mathbf{v}} = (\mathbf{A} - \sigma_j \mathbf{B})^{-1}\mathbf{B}\mathbf{w}_j$ are nonzero and thus so do the first $j+1$ ones of $\mathbf{v}_{j+1}$. This completes the proof. □

An important consequence of Lemmas 3.5–3.6 is that, at each iteration j of the rational Krylov method, we only use the submatrices $\mathbf{A}_j$ and $\mathbf{B}_j$ for computing the next vector $\mathbf{v}_{j+1}$. Consequently, we only use the interpolation nodes $\sigma_0, \ldots, \sigma_j$ and the corresponding divided difference matrices $A_0, \ldots, A_j$.

Proposition 3.7. *The Ritz values λ_i, computed at iteration j of the rational Krylov method are independent of d as long as $j < d$. These Ritz values are also independent of $\sigma_{j+1}, \ldots, \sigma_d$.*

Proof. At iteration j, the Ritz values are computed from the upper $j \times j$ parts of the two Hessenberg matrices $\underline{H}_j$ and $\underline{K}_j$, which are obtained from the orthogonalization process of only $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{j+1}$. Following Lemmas 3.5–3.6, only the first $j+1$ interpolation points $\sigma_0, \ldots, \sigma_j$ are used for the construction of the rational Krylov subspace $\mathbf{V}_{j+1}$. Therefore, the approximated eigenvalues are independent of the interpolation points $\sigma_{j+1}, \ldots, \sigma_d$. Hence they are also independent of d, which proves the proposition. □

Corollary 3.8. *It is neither necessary to choose the interpolation nodes σ_j in advance, nor the degree d of the interpolating polynomial $P(\lambda)$. Instead, in each iteration we can choose the next interpolation node based on the results of the previous iterations. Therefore, the rational Krylov method can be implemented in an adaptive and an incremental way. The rational Krylov method is initialized with an interpolation node σ_0 and a particular starting vector, and can run until convergence by dynamically adding a node σ_j in each iteration.*

Remark 3.9. Performing j steps of our method, with an appropriate starting vector, produces the same subspace as j steps of the standard rational Krylov method with the matrices $\mathbf{A}$ and $\mathbf{B}$ for any $d > j$. Taking a limit argument, $d \to \infty$, the Newton Rational Krylov method can be interpreted as a rational Krylov method directly applied to an infinite dimensional linear problem equivalent to the original nonlinear problem. As in the Taylor–Arnoldi method, only finite arithmetic is used.

Remark 3.10. Another consequence of the independence of d is that we can restart the rational Krylov algorithm without any adaptation. Since at any iteration j, the method is independent of $\sigma_{j+1}, \sigma_{j+2}, \ldots$, we can return to iteration $k < j$ and continue from this iteration with other interpolation points $\sigma_{k+1}, \sigma_{k+2}, \ldots$.

3.2.3 Algorithm

Based on Lemmas 3.5–3.6 and the important Corollary 3.8, the Newton Rational Krylov algorithm for solving the NLEP (3.1) can be dynamically implemented. Algorithm 3.2 gives an outline and Figure 3.2 shows a visualization. As in the Taylor–Arnoldi method, we can subdivide each iteration j of Algorithm 3.2 into two phases: an expansion phase followed by a rational Krylov phase.

Algorithm 3.2: Newton Rational Krylov method [106]

1 Choose shift σ_0 and starting vector $v_1 \in \mathbb{C}^n$.
for $j = 1, 2, \ldots$ **do**
  EXPANSION PHASE:
2 Choose shift: σ_j.
3 Compute next divided difference matrix: A_j.
4 Expand $\mathbf{A}_j$, $\mathbf{B}_j$ and $\mathbf{V}_j$.
  RATIONAL KRYLOV PHASE:
5 Set continuation combination: t_j.
6 Compute $\widehat{\mathbf{v}} := (\mathbf{A}_j - \sigma_j \mathbf{B}_j)^{-1} \mathbf{B}_j \mathbf{V}_j t_j$.
7 Orthogonalize: $\widetilde{\mathbf{v}} := \widehat{\mathbf{v}} - \mathbf{V}_j h_j$, where $h_j = \mathbf{V}_j^* \widehat{\mathbf{v}}$.
8 Get new vector: $\mathbf{v}_{j+1} = \widetilde{\mathbf{v}} / h_{j+1,j}$, where $h_{j+1,j} = \|\widetilde{\mathbf{v}}\|$.
9 Compute Ritzpairs: $(\lambda_i, \mathbf{x}_i)$.
10 Nonlinear eigenpairs: $\left(\lambda_i, x_i^{[1]}\right)$ and test for convergence.
end

Firstly, in the expansion phase (lines 2–4), we choose at iteration j the next interpolation node σ_j and compute the corresponding divided difference matrix A_j in order to extend the linearization matrices $\mathbf{A}_{j-1}$ and $\mathbf{B}_{j-1}$ to $\mathbf{A}_j$ and $\mathbf{B}_j$, respectively. We also extend the matrix $\mathbf{V}_j$ with a zero block at the bottom.

Secondly, in the rational Krylov phase (lines 5–9), we perform a rational Krylov step with the extended matrices $\mathbf{A}_j$ and $\mathbf{B}_j$, and shift σ_j. In line 9, the Ritz values are computed in the same way as in the standard rational Krylov method (Algorithm 1.3), i.e., $(\lambda_i, \mathbf{x}_i)$ is a Ritzpair of $(\mathbf{A}, \mathbf{B})$, with

$$K_j s_i = \lambda_i H_j s_i, \qquad s_i \neq 0,$$

and $\mathbf{x}_i = \mathbf{V}_{j+1} \underline{H}_j s_i$.

Finally, in line 10, we take the first blocks of $\mathbf{x}_i$ as approximations for the nonlinear eigenvectors and check for convergence of the nonlinear eigenpairs $\left(\lambda_i, x_i^{[1]}\right)$. For more details about how to select the nonlinear eigenvector we refer to [49, Section 3.2].

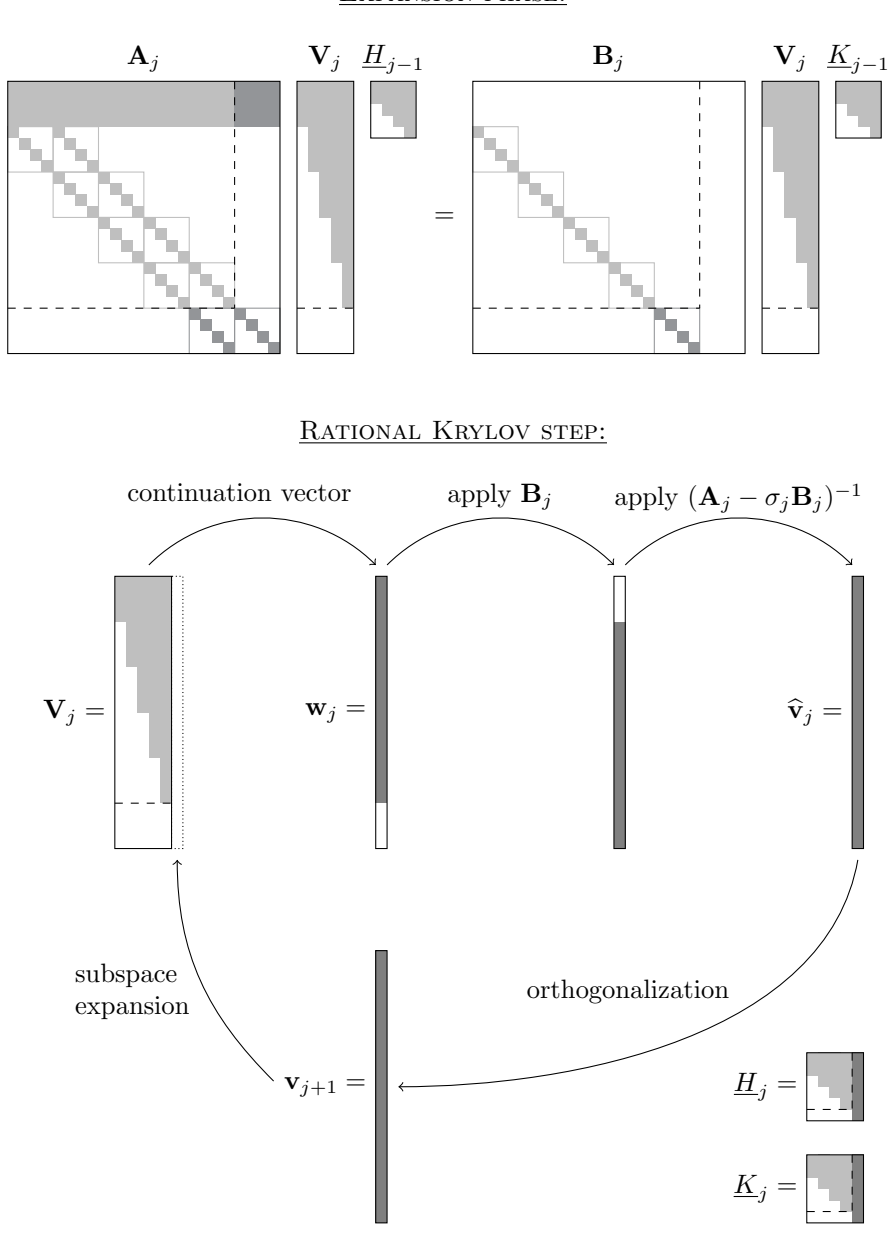


Figure 3.2: Visualization of Algorithm 3.2 [106].

Computing divided differences

We start with the property that a matrix-valued function $A(\lambda) \in \mathbb{C}^{n\times n}$ can always be written in the following form

$$A(\lambda) = \sum_{k=1}^{m} B_k f_k(\lambda), \tag{3.16}$$

where $B_k \in \mathbb{C}^{n\times n}$ are constant matrices, f_k are scalar functions of λ and $m \leq n^2$. However, in many applications, such as e.g. time-delay eigenvalue problems, $m \ll n$. Other applications are mathematical models that are linear in λ but adopt nonlinear boundary conditions, which lead to low rank B_k and $m \ll n$.

Let $A(\lambda)$ be given in the form (3.16) and suppose that the scalar functions f_k are approximated by scaled interpolating Newton polynomials

$$f_k(\lambda) \approx \sum_{i=0}^{d} \alpha_{ik} n_i(\lambda), \qquad k = 1, \ldots, m, \tag{3.17}$$

where α_{ik} are the scaled divided differences and n_i are the scaled Newton basis function defined in (3.10). Note that these scaled divided differences can be computed by the following variant of the Optiz theorem.

Theorem 3.11. *The scaled divided differences $\alpha_0, \ldots, \alpha_d$ of the scaled interpolating Newton matrix polynomial are the elements in the first row of $\beta_0 f(M)$, where*

$$M = \begin{bmatrix} \sigma_0 & \beta_1 & & & \\ & \sigma_1 & \beta_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & \beta_d \\ & & & & \sigma_d \end{bmatrix}.$$

Next, substituting (3.17) in (3.16) yields the following (scaled) interpolating Newton matrix polynomial

$$A(\lambda) \approx P(\lambda) := \sum_{k=1}^{m} B_k \sum_{i=0}^{d} \alpha_{ik} n_i(\lambda) = \sum_{i=0}^{d} \left(\sum_{k=1}^{m} \alpha_{ik} B_k \right) n_i(\lambda). \tag{3.18}$$

Combining now (3.9) and (3.18) results in

$$A_i = \sum_{k=1}^{m} \alpha_{ik} B_k,$$

which means that the divided differences matrices A_i are linear combinations of the constant coefficient matrices B_k. Consequently, in step 3 of Algorithm 3.2 we only need to compute and store the scalar divided differences α_{ik}.

Breakdown

Breakdown of the rational Krylov method leads to an invariant subspace. However, possible breakdown of Algorithm 3.2 needs some attention and differs from breakdown in Algorithm 1.3.

Proposition 3.12. *If $v_1 \neq 0$, then the shifts in Algorithm 3.2 can always be chosen such that the Gram–Schmidt orthogonalization process never causes a breakdown.*

Proof. A breakdown of the orthogonalization process in iteration j of Algorithm 3.2 occurs when the range($\mathbf{V}_j$) is an invariant subspace. In other words, this can only be the case when the $(j+1)$th block of $\mathbf{v}_{j+1}$ is zero. By using $\sigma_j = \sigma_{j-1}$ in iteration j of Algorithm 1.3, it follows that the $(j+1)$th block of $\mathbf{v}_{j+1}$ is nonzero. This proves the proposition. □

However, when we choose the continuation vector as defined in (1.16) breakdown does not happen in practice. During all the numerical experiments we performed, no breakdown occurred.

3.2.4 Low rank exploitation

In several applications the NLEP (3.1) consists of a polynomial part and a nonlinear part which is of low rank. See for example the `'gun'` problem and the `'particle in a canyon'` problem discussed in §1.1.2. This low rank structure can be exploited by using a different type of linearization [106].

Suppose that the NLEP is defined as follows

$$A(\lambda)x = \left(\sum_{i=0}^{p} B_i \lambda^i + \sum_{i=1}^{m} C_i f_i(\lambda) \right) x = 0, \tag{3.19}$$

where $B_i, C_i \in \mathbb{C}^{n\times n}$ are constant matrices, $f_i(\lambda)$ are scalar functions of λ, $p \ll n^2$ and $m \ll n^2$. Furthermore, we assume that the matrices C_i have rank-revealing factorizations $C_i = L_i U_i^*$, where $L_i, U_i \in \mathbb{C}^{n\times r_i}$ are of full column rank $r_i \ll n$.

Approximating the scalar functions $f_i(\lambda)$ of (3.19) by interpolating polynomials in Newton form results in the following matrix polynomial which interpolates $A(\lambda)$ at the interpolation points $\sigma_0, \sigma_1, \ldots, \sigma_d$

$$\widetilde{P}(\lambda) = \sum_{i=0}^{d} \widetilde{A}_i n_i(\lambda) = \sum_{i=0}^{p} \left(\widetilde{B}_i + \widetilde{C}_i \right) n_i(\lambda) + \sum_{i=p+1}^{d} \widetilde{C}_i n_i(\lambda), \tag{3.20}$$

where

$$\widetilde{B}_i = \sum_{j=0}^{p} \beta_{ij} B_j, \qquad \text{and} \qquad \widetilde{C}_i = \sum_{j=1}^{m} \gamma_{ij} C_j = \sum_{j=1}^{m} \gamma_{ij} L_j U_j^*, \tag{3.21}$$

with scalars β_{ij} and γ_{ij}. Define

$$\begin{aligned} \widetilde{L}_i &= \begin{bmatrix} \gamma_{i1} L_1 & \gamma_{i2} L_2 & \cdots & \gamma_{im} L_m \end{bmatrix}, \qquad i = 1, \ldots, m, \\ \widetilde{U} &= \begin{bmatrix} U_1 & U_2 & \cdots & U_m \end{bmatrix}, \end{aligned}$$

where the size of $\widetilde{L}_i$ and $\widetilde{U}$ is $n \times r$ and $r = r_1 + r_2 + \cdots + r_m$.

Similarly as in Theorem 2.13, we obtain a companion-type reformulation where the pair $(\lambda, x \neq 0)$ is an eigenpair of the polynomial eigenvalue problem (3.20) if and only if

$$\widetilde{\mathbf{A}}\widetilde{\mathbf{x}} = \lambda \widetilde{\mathbf{B}} \widetilde{\mathbf{x}},$$

where

$$\widetilde{\mathbf{A}} = \begin{bmatrix} \widetilde{A}_0 & \widetilde{A}_1 & \cdots & \widetilde{A}_p & \widetilde{L}_{p+1} & \cdots & \widetilde{L}_{d-2} & \widetilde{L}_{d-1} - \sigma_{d-1}\widetilde{L}_d/\beta_d \\ \sigma_0 I & \beta_1 I & & & & & & \\ & \ddots & \ddots & & & & & \\ & & \sigma_{p-1} I & \beta_p I & & & & \\ & & & \sigma_p \widetilde{U}^* & \beta_{p+1} I & & & \\ & & & & \sigma_{p+1} I & \beta_{p+2} I & & \\ & & & & & \ddots & \ddots & \\ & & & & & & \sigma_{d-2} I & \beta_{d-1} I \end{bmatrix}$$

with $\widetilde{A}_i = \widetilde{B}_i + \widetilde{C}_i$ for $i = 0, 1, \ldots, p$ and

$$\widetilde{\mathbf{B}} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & -\widetilde{L}_d/\beta_d \\ I & 0 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & I & 0 & & & & \\ & & & \widetilde{U}^* & 0 & & & \\ & & & & I & 0 & & \\ & & & & & \ddots & \ddots & \\ & & & & & & I & 0 \end{bmatrix}, \qquad \widetilde{\mathbf{x}} = \begin{bmatrix} n_0(\lambda) x \\ n_1(\lambda) x \\ \vdots \\ n_p(\lambda) x \\ n_{p+1}(\lambda) \widetilde{U}^* x \\ \vdots \\ b_{d-2}(\lambda) \widetilde{U}^* x \\ b_{d-1}(\lambda) \widetilde{U}^* x \end{bmatrix}.$$

For this type of linearization we can also adapt Lemmas 3.5–3.6.

3.2.5 Connection with Newton's method

We are now ready to discuss a connection of Algorithm 3.2 with Newton's method. In each iteration of Newton's method, the NLEP (3.1) can be written as follows

$$A(\lambda_{j+1})x_{j+1} = A(\lambda_j + \Delta\lambda_j)(x_j + \Delta x_j) \approx 0,$$

where $\Delta\lambda_j = \lambda_{j+1} - \lambda_j$ and $\Delta x_j = x_{j+1} - x_j$. Using a first order approximation of $A(\lambda_j + \Delta\lambda_j)$ results in

$$[A(\lambda_j) + A'(\lambda_j)\Delta\lambda_j](x_j + \Delta x_j) \approx 0, \tag{3.22}$$

and by omitting the higher order term, $O(\Delta\lambda_j \Delta x_j)$, we deduce

$$A(\lambda_j)(x_j + \Delta x_j) + A'(\lambda_j)x_j\Delta\lambda_j \approx 0. \tag{3.23}$$

Using (3.23) we define

$$x_{j+1} := \frac{v_j}{\|v_j\|}, \qquad \text{with} \qquad v_j = -A(\lambda_j)^{-1}A'(\lambda_j)x_j.$$

Note that the factor $\Delta\lambda_j$ cancels out due to the normalization. By multiplying (3.22) on the left with x_{j+1}^*, we find the Newton update for the approximate eigenvalue

$$\lambda_{j+1} := \lambda_j - \frac{x_{j+1}^* A(\lambda_j)x_{j+1}}{x_{j+1}^* A'(\lambda_j)x_{j+1}}.$$

The connection between the linear rational Krylov algorithm and the Jacobi–Davidson algorithm [94] is illustrated in [85]. From [95], we also know that each step of the Jacobi–Davidson iteration method can be interpreted as a Newton update. Since the rational Krylov method is a subspace method, we expect that using Ritz values as shifts reaches asymptotically at least quadratic convergence.

As already mentioned in Remark 3.9, Algorithm 3.2 can also be interpreted as a standard linear rational Krylov method applied to the matrices $\mathbf{A}$ and $\mathbf{B}$, which are obtained from a linearization with the shifts as interpolation points. Using Ritz values as shifts, $A'(\lambda_j)$ is approximated better and better in each iteration. In the real case, using the mean value theorem for divided differences, the approximation error vanishes exponentially, since

$$A'(\lambda_j) - P_j'(\lambda_j) = O\big(n_j(\lambda_j)\big) = O\big((\lambda_j - \lambda_0)(\lambda_j - \lambda_1)\cdots(\lambda_j - \lambda_{j-1})\big).$$

Therefore, we also expect an asymptotically (super) quadratic convergence for Algorithm 3.2. This is illustrated in §3.3.2.

3.3 Numerical experiments

We illustrate the Taylor–Arnoldi method (Algorithm 3.1) and the Newton Rational Krylov method (Algorithm 3.2) with two numerical examples. Firstly, we consider a root-finding problem in §3.3.1. For this scalar nonlinear eigenvalue problem, we show the difference between Taylor approximations used in the Taylor–Arnoldi method and Newton interpolation in the Newton Rational Krylov method. For the latter, we make a distinction between Newton interpolation in Leja points [64] and Hermite interpolation. Next, we consider the `'gun'` problem discussed in §1.1.2. For this large-scale nonlinear eigenvalue problem, we first perform a global eigenvalue search with the Newton Rational Krylov method and compare to the Taylor–Arnoldi method; we then use the Newton Rational Krylov method for local correction and compare to Newton's method.

All numerical experiments are performed in MATLAB version 7.14.0 (R2012a) on a Dell Latitude notebook running an Intel(R) Core(TM) i5-2540M CPU @ 2.60GHz quad core processor with 8 GB RAM. Our experiments can be reproduced with the publicly available code of [44] and [107].

3.3.1 Root-finding problem

We start with the following scalar NLEP

$$A(\lambda) = e - \frac{3}{4} - 3\lambda + \left(\lambda + \frac{5}{4}\right)^2 - e^{\lambda+\frac{1}{4}} - e^{\frac{3}{4}-\lambda} = 0, \tag{3.24}$$

and are interested in the real roots in the interval $[-1.25, 1.25]$, which are $\lambda_1 = -0.25$ and $\lambda_2 = 0.75$.

Remark 3.13. All the eigenvectors are equal to 1 in the scalar case. Since $n = 1$, matrix $\mathbf{V}$, which is constructed in Algorithm 3.1 or Algorithm 3.2, is the identity matrix. Therefore, it is only necessary to store the Hessenberg matrices H and K from which the approximate eigenvalues are computed.

Remark 3.14. If Algorithm 3.1 or Algorithm 3.2 is used to find the roots of a polynomial of degree k, then after k iterations the k roots are found as Ritz values. Moreover, since breakdown can always be avoided, see Proposition 3.12, additional iterations produce additional roots, which are all infinite.

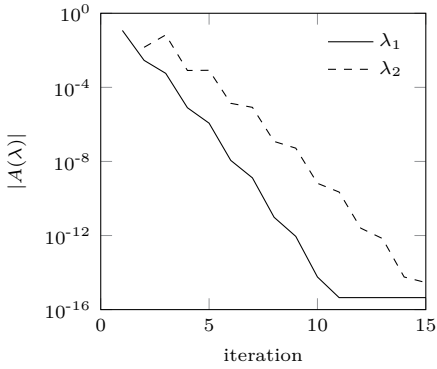
We now compare the Taylor–Arnoldi method and the Newton Rational Krylov method. For the latter, we make a distinction between Newton interpolation in Leja points and Hermite interpolation.

Taylor approximation

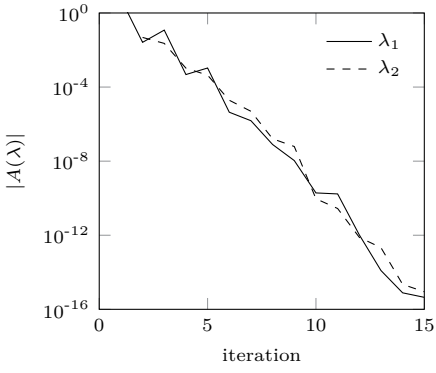
In the Taylor–Arnoldi method, the nonlinear function (3.24) is approximated by truncated Taylor series with shift $\sigma = 0$. This results in a fast convergence

of the approximation close to the origin and a slower convergence at the ends of the interval of interest.

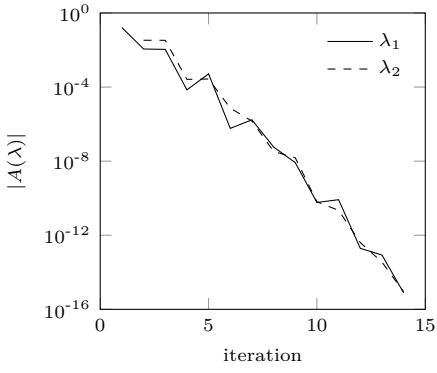
The convergence history for the eigenvalues λ_1 and λ_2 is shown in Figure 3.3(a). From this figure, we see that λ_1 starts very quickly to converge since this eigenvalue is closely located to the origin. We also note that after some more iterations the eigenvalue λ_2 starts to converge. This happens when the approximation by the truncated Taylor series is also improving in the neighborhood of this eigenvalue.



(a) Taylor approximation



(b) Newton interpolation in Leja points



(c) Hermite interpolation

Figure 3.3: Root finding problem: convergence history for λ_1 (solid line) and λ_2 (dashed line) of the scalar NLEP. (a) Taylor approximation, (b) Newton interpolation in Leja points in the interval $[-1.25, 1.25]$, and (c) Hermite interpolation in the points -1, 0, and 1, all with multiplicity 5.

Newton interpolation in Leja points

A Taylor approximation has the disadvantage that its convergence is not uniform on the whole interval. To overcome this we can use the Newton Rational Krylov method.

A first possible technique for selecting the shifts is choosing the shifts in Leja fashion [30, 64, 81, 10] in the interval of interest. Leja points have the property that their limit distribution on an interval is the same as the limit distribution of the zeros of shifted and scaled Chebyshev polynomials for the same interval.

The convergence history for λ_1 and λ_2, computed by Algorithm 3.2 with Leja points in the interval $[-1.25, 1.25]$ is shown in Figure 3.3(b). This figure illustrates that after some iterations the eigenvalues start to converge. Note that the convergence of λ_1 is now a bit slower than in the Taylor–Arnoldi method. However, by making use of polynomial interpolation in Leja points, we obtain a more uniform convergence history for all the eigenvalues in the interval of interest.

Hermite interpolation

Another possibility in the Newton Rational Krylov method is using Hermite interpolation in a few points chosen in the interval of interest. Here, we chose the interpolation points -1, 0, and 1, all with multiplicity 5.

The corresponding convergence history for λ_1 and λ_2, computed by Algorithm 3.2 with cyclically repeated interpolation points -1, 0, and 1, is shown in Figure 3.3(c). This figure shows that we have again a uniform convergence history for all the eigenvalues in the interval of interest. Moreover, for large-scale NLEPs, Hermite interpolation has the advantage compared to interpolation in Leja points that we need to compute less LU-factorizations.

3.3.2 Gun problem

We consider the `'gun'` problem, see §1.1.2, of the NLEVP collection [14]. This is a large-scale NLEP that models a radio-frequency gun cavity and is of the form

$$A(\lambda)x = \left(K - \lambda M + i\sqrt{\lambda - \sigma_1^2}W_1 + i\sqrt{\lambda - \sigma_2^2}W_2\right)x = 0, \tag{3.25}$$

where M, K, W_1 and W_2 are real symmetric matrices of size 9956×9956, K is positive semidefinite, M is positive definite, and $\operatorname{rank}(W_1) + \operatorname{rank}(W_2) = 84$. As in [14], we take $\sigma_1 = 0$ and $\sigma_2 = 108.8774$. The notation of the complex

square root, $\sqrt{\cdot}$, denotes the principal branch. The domain of interest is such that $\mathrm{Im}(\lambda) \geq 0$ and $\mathrm{Re}(\lambda)$ is bounded away from the branch points $\lambda = 0$ and $\lambda = \sigma_2^2$ [68].

As in [52], we first shifted and scaled the original problem (3.25) by $\lambda = \gamma\widehat{\lambda} + \mu$, such that the region of interest was transformed to be roughly within the unit circle. Therefore, we chose $\gamma = 300^2 - 200^2$ and $\mu = 250^2$. We obtained the following transformed NLEP

$$\left(K - (\gamma\widehat{\lambda} + \mu)M + i\sqrt{\gamma\widehat{\lambda} + \mu - \sigma_1^2}W_1 + i\sqrt{\gamma\widehat{\lambda} + \mu - \sigma_2^2}W_2\right)x = 0, \quad (3.26)$$

which was solved by Algorithm 3.2.

For measuring the convergence of an approximate eigenpair (λ, x), we used, as defined in [68], the relative residual norm

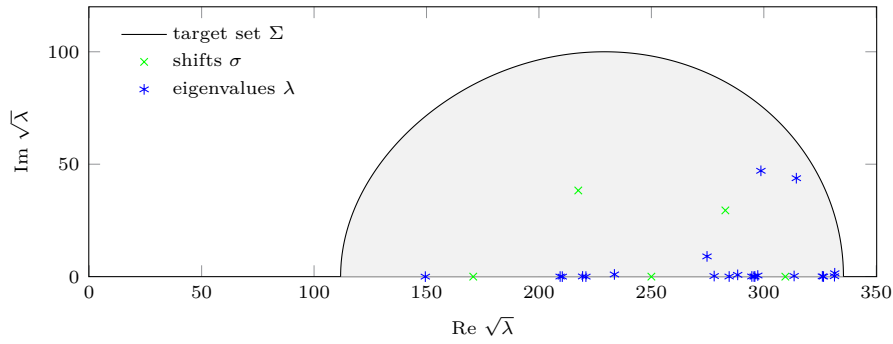
$$E(\lambda, x) = \frac{\|A(\lambda)x\|_2/\|x\|_2}{\|K\|_1 + |\lambda|\|M\|_1 + \sqrt{|\lambda - \sigma_1^2|}\|W_1\|_1 + \sqrt{|\lambda - \sigma_2^2|}\|W_2\|_1}.$$

Global eigenvalue search

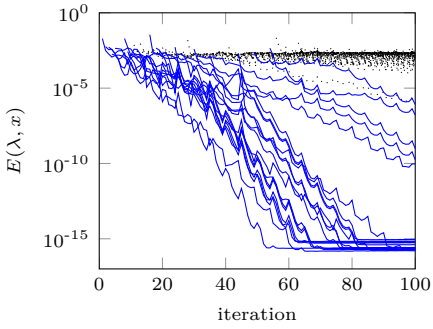
In this paragraph, we illustrate the Newton Rational Krylov method as global search method for the `'gun'` problem. Since the domain of interest of the transformed NLEP (3.26) is roughly the top half part of the unit circle, we chose 5 interpolation points almost uniformly distributed in this half circle. These interpolation points are indicated by "$\times$" in Figure 3.4(a) and are cyclically repeated in Algorithm 3.2.

The convergence history of the eigenpairs computed by Algorithm 3.2 is given in Figure 3.4(b). Note that the solid and dotted lines correspond to eigenvalues lying inside and outside the target set, respectively. The square roots of the corresponding 21 approximate eigenvalues, which are lying in the domain of interest, are shown in Figure 3.4(a). We repeated this experiment with the low rank exploiting version of Algorithm 3.2, discussed in §3.2.4. The corresponding convergence history is shown in Figure 3.4(c) and is very similar to the convergence history of the standard implementation of Algorithm 3.2, shown in Figure 3.4(b). However, the total memory usage for the subspace $\mathbf{V}$ is significantly reduced.

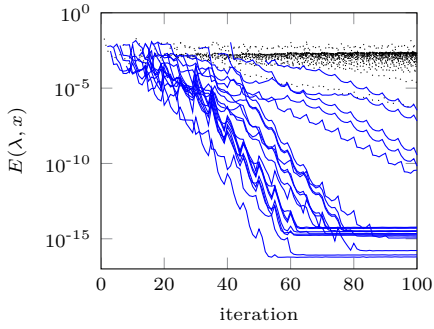
A comparison of Algorithm 3.2 and its low rank exploiting version are given in Table 3.1. From this table follows that the significant reduction in computation time of the low rank version is due to the orthogonalization process. Indeed, in each iteration $j > p$ of the low rank exploiting version, the vectors have dimension $(p+1)n + (j-p)r$. On the other hand, in the version without low rank exploiting, the vectors are of dimension $(j+1)n$. In this problem



(a) shifts/interpolation points and eigenvalues



(b) Algorithm 3.2



(c) Algorithm 3.2 with low rank exploitation

Figure 3.4: Results for the 'gun' problem: (a) Approximate eigenvalues for the original NLEP (3.25) obtained with Algorithm 3.2, (b) convergence history for Algorithm 3.2, and (c) convergence history for Algorithm 3.2 with low rank exploitation.

Table 3.1: Timings for the 'gun' problem.

Operation	Algorithm 3.2	Algorithm 3.2 + low rank
LU decompositions	5.3 s	5.3 s
Linear system solves	2.4 s	2.4 s
Gram–Schmidt orth. + reorth.	29.7 s	1.2 s
Total	40.7 s	8.9 s

$n = 9956$, $p = 1$ and $r = 84$ which explains the low computational cost of the orthogonalization process in the low rank exploiting version of Algorithm 3.2. For further reduction of the memory and orthogonalization cost, we refer to Chapter 6.

To illustrate the importance of using Hermite interpolation in the global eigenvalue search strategy, we also solved the transformed NLEP (3.26) with the Taylor–Arnoldi method (Algorithm 3.1) for different shifts. Table 3.2 gives the number of eigenvalues with relative residual norm $E(\lambda, x) \leq 10^{-6}$. This table indicates that the number of accurately computed eigenvalues strongly depends on the choice of shift σ and is always smaller than the number of eigenvalues computed by Algorithm 3.2 with Hermite interpolation, see Figure 3.4(a).

Table 3.2: Taylor–Arnoldi method versus Newton Rational Krylov method for the `'gun'` problem. The number of accurate eigenvalues, $\#\lambda$, obtained after 100 iterations of Algorithm 3.1 with different shifts σ. Note that the number between parentheses is the number of accurate eigenvalues outside the target set. The last line shows the results obtained after 100 iterations of Algorithm 3.2 with Hermite interpolation in the points indicated in Figure 3.4(a).

σ	$\#\lambda : E(\lambda, x) \leq 10^{-6}$
$-2/3$	6
$-1/3 + i/3$	11
0	17
$1/3 + i/3$	20 (+3)
$2/3$	16 (+7)
Hermite	21 (+1)

Thus, we can conclude that the Newton Rational Krylov method with Hermite interpolation is really suitable for finding eigenvalues in a specified region of interest. Furthermore, since this method uses an approximation of the nonlinear matrix-valued function $A(\lambda)$ based on Hermite interpolation, this method is more suitable for global eigenvalue search than the Taylor–Arnoldi method, which uses only one interpolation point.

Local eigenvalue correction

The numerical experiments of the previous paragraph have shown that Algorithm 3.2 performs well as a global method. In this paragraph, we now illustrate that the Newton Rational Krylov method can also be used as a local correction method and we compare to Newton's method.

Therefore, we return to the square root `'gun'` problem (3.25). As suggested in [14], we consider the eigenvalue λ for which $\sqrt{\lambda}$ is nearest to 146.71. We first

performed Newton's method, outlined in §3.2.5, with $\lambda_0 = 146.71^2$ and x_0 a random vector. During the first iteration we took $\lambda_1 = \lambda_0$, since otherwise the algorithm would converge to an eigenvalue outside the region of interest. This could be explained by the fact that the random starting vector x_0 was not yet a good approximation of the eigenvector corresponding to the eigenvalue nearest to 146.71^2. The resulting convergence history towards the eigenvalue nearest to λ_0 is shown in Table 3.3(a).

Table 3.3: Convergence history of the approximate eigenvalue of the 'gun' problem nearest to 146.71^2 with (a) Newton's method and with (b) the Newton Rational Krylov method.

(a) Newton's method

i	$\sqrt{\lambda}$	$E(\lambda, x)$
0	146.71	4.8544e-02
1	146.71	5.7677e-04
2	$149.10 + 0.012i$	3.5515e-05
3	$149.48 + 0.002i$	1.8332e-07
4	$149.48 + 0.002i$	6.0417e-14
5	$149.48 + 0.002i$	1.3171e-17

(b) Newton Rational Krylov method

i	$\sqrt{\sigma}$	$\sqrt{\lambda}$	$E(\lambda, x)$
0	146.71		4.8544e-02
1	146.71	$153.13 + 0.007i$	2.7855e-05
2	$153.13 + 0.007i$	$149.48 + 0.002i$	1.5795e-07
3	$149.48 + 0.002i$	$149.48 + 0.002i$	2.6212e-12
4	$149.48 + 0.002i$	$149.48 + 0.002i$	5.0740e-16

Now, we compare the Newton Rational Krylov method to Newton's method. To this end, we started Algorithm 3.2 with $\sigma_0 = \sigma_1 = 146.71^2$. For the other shifts, we chose, in each iteration, the Ritz value of the previous iteration which resulted in the smallest relative residual norm. The corresponding convergence history of the rational Krylov method is shown in Table 3.3(b). From this table, we can conclude that the Newton Rational Krylov method converges in less iterations than Newton's method. This is expected because the rational Krylov method is a subspace method which builds an expanding subspace.

Recall a second difference between the Newton Rational method and Newton's method. The first one can converge at the same time to more than one eigenvalue.

3.4 Conclusions

In this chapter, we introduced the idea of dynamic interplay between polynomial interpolation, linearization, and (rational) Krylov methods for solving the nonlinear eigenvalue problem.

Firstly, in the Taylor–Arnoldi method we used truncated Taylor series, i.e., Hermite interpolation in only one point, to approximate the matrix-valued function $A(\lambda)$. Using the property that adding a new term to the Taylor series only adds one block row and column to the linearization pencil, while keeping the remaining part unchanged, together with solving the obtained generalized linear eigenvalue problem by Arnoldi's method with a short starting vector, results in a dynamical algorithm where the dimension of the pencil grows in every iteration. Moreover, by exploiting its structure only standard linear algebra operations with matrices of the original NLEP dimension are involved.

Secondly, in the Newton Rational Krylov method we used Newton/Hermite interpolation to approximate $A(\lambda)$ which results in a better approximation than a truncated Taylor series of the same degree. As in the monomial basis, adding a new interpolation point does not affect the previous linearization pencil and only adds one block row and column. Solving this pencil with a rational Krylov method with a short starting vector and where the shifts are no free parameters but equal to the interpolation points yields again a dynamic algorithm with a growing pencil in every iteration. Furthermore, neither the degree of the interpolating polynomial, nor the interpolation points need to be fixed in advance.

Is this dynamic polynomial interpolation possible in every basis? To answer this question we need to have a closer look on which properties the dynamic interplay between linearization and (rational) Krylov methods rely: (i) the property that the linearization matrices of lower order matrix polynomials are submatrices of the higher order ones and (ii) the property that the system decouples by using appropriate shifts and that the "trick" with a short starting vector can be applied. Although any polynomial has infinitely many linearizations, only the companion linearization in monomial and Newton basis of Chapter 2 satisfy both properties. The interpolating matrix polynomial in Lagrange form, although very easy to construct, violates the first property. Increasing the degree of the interpolating polynomial by adding a new interpolation point can be done efficiently. However, in this case all the basis functions will change completely and also do the linearization matrices except for the first block row. Furthermore, the companion linearization pencils of interpolating matrix polynomials in any orthogonal basis have always a second nonzero block subdiagonal. Therefore, the second property is violated and the "trick" with the short starting vector cannot be applied.

Chapter 4

Rational Interpolation

We consider the nonlinear eigenvalue problem (NLEP)

$$A(\lambda)x = 0, \tag{4.1}$$

where $\lambda \in \Omega \subseteq \mathbb{C}$ and $x \in \mathbb{C}^n \setminus \{0\}$. In this chapter, we aim to compute eigenvalues in a compact *target set* $\Sigma \subset \Omega$. Therefore, we assume that the matrix-valued function $A : \Sigma \to \mathbb{C}^{n\times n}$ depends analytically on λ, i.e., each component of $A(\lambda)$ is an analytic function of λ.

For methods relying on *polynomial interpolants* of $A(\lambda)$, the convergence is limited by the convergence of polynomials. With an appropriate choice of the interpolation nodes, polynomial interpolation performs very well if $A(\lambda)$ is an entire function, in which case any polynomial interpolant is guaranteed to converge throughout the complex plane, or when singularities of $A(\lambda)$ are sufficiently far away from the target set Σ. However, if $A(\lambda)$ is difficult to approximate by polynomials, the performance of the (rational) Krylov methods will be limited by the accuracy of the underlying polynomial expansion. Therefore, we propose a new rational Krylov method where the nonlinearity $A(\lambda)$ is explicitly expanded in *rational functions* of λ, hence the name Fully Rational Krylov method. The combination of a linearization, based on rational Newton basis functions, with a rational Krylov method offers a lot of flexibility.

This chapter is organised as follows. Firstly, we give a motivating example in Section 4.1 showcasing the possible advantages of rational over polynomial interpolation. Next, we introduce in Section 4.2 the Fully Rational Krylov method which is based on dynamic linear rational interpolation in Newton basis interwoven with the rational Krylov method. In Section 4.3 the different variants of the Fully Rational Krylov method are illustrated with several numerical examples. Finally, we summarise the main results in Section 4.4.

4.1 Motivating example

We now give some motivation for the approach of the chapter by means of the simple scalar NLEP

$$A(\lambda) = 0.2\sqrt{\lambda} - 0.6\sin(2\lambda), \tag{4.2}$$

which illustrates the improvements that can potentially be achieved by using a rational expansion instead of a polynomial expansion. Suppose we want to find all 3 real eigenvalues of $A(\lambda)$ on the interval $\Sigma = [\alpha, \beta] = [10^{-2}, 4]$. Of course, this is in fact just a root finding problem (see Figure 4.1).

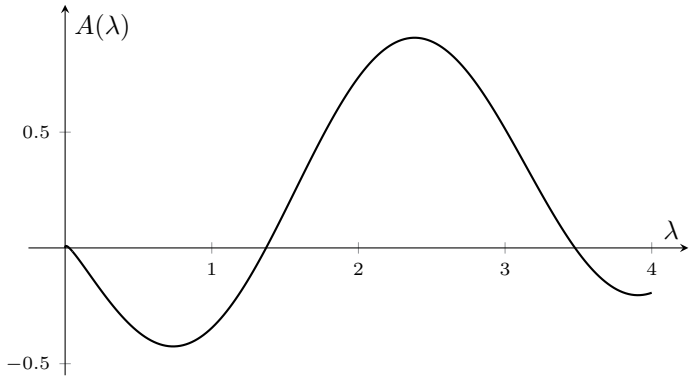


Figure 4.1: Scalar root finding example.

A natural solution approach is to first approximate $A(\lambda)$ by a polynomial interpolant $P(\lambda)$ of degree d, interpolating at nodes $\sigma_0, \sigma_1, \ldots, \sigma_d \in \Sigma$, and then to compute the roots of $P(\lambda)$ on Σ. See [17] for a recent review of polynomial root finding. Let $\lambda_\star$ be such a root, i.e., $P(\lambda_\star) = 0$. Then the residual $A(\lambda_\star)$ is bounded by the uniform approximation error of $P(\lambda)$ for $A(\lambda)$:

$$|A(\lambda_\star)| \leq \max_{\lambda\in\Sigma} |A(\lambda) - P(\lambda)| =: \big\|A(\lambda) - P(\lambda)\big\|_\Sigma.$$

In view of this inequality, it is natural to make the error $\|A(\lambda) - P(\lambda)\|_\Sigma$ small. The asymptotic convergence of $P(\lambda)$ to $A(\lambda)$ in the uniform norm is determined by the location of the singularities of $A(\lambda)$ relative to Σ, and by the distribution of interpolation nodes on Σ. For interpolation nodes which are chosen asymptotically optimal on Σ we can show that convergence takes place at a geometric rate

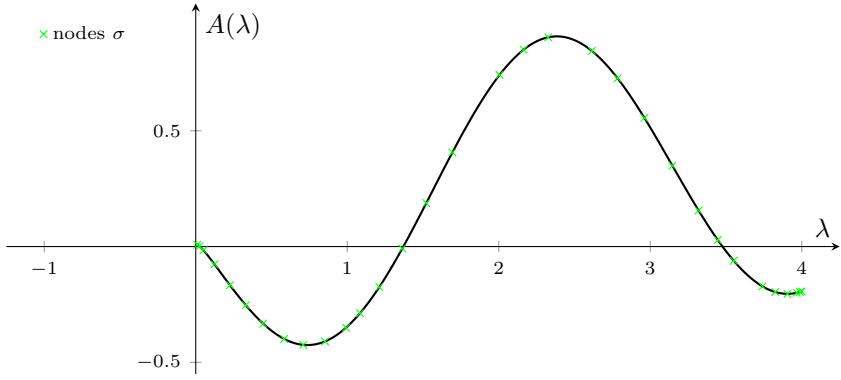
$$\limsup_{d\to\infty} \big\|A(\lambda) - P(\lambda)\big\|_\Sigma^{1/d} \leq \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right) \lesssim \exp\left(-\frac{2}{\sqrt{\kappa}}\right), \tag{4.3}$$

with $\kappa = \beta/\alpha$ and $\lesssim$ denoting an approximate upper bound that is asymptotically sharp for large κ. This is the best possible asymptotic convergence that can be achieved by polynomial interpolation. Examples of interpolation nodes for which this convergence is achieved are the Chebyshev points

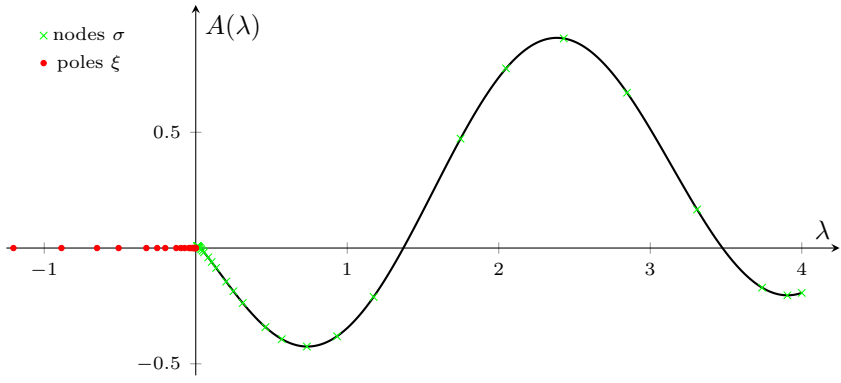
$$\sigma_j = \frac{\alpha+\beta}{2} + \frac{\alpha-\beta}{2}\cos(\pi j/d), \qquad j = 0, 1, \ldots, d, \tag{4.4}$$

and *Leja points* [81, 10]. The latter have the property that their limit distribution on an interval is the same as the limit distribution of (4.4). Figure 4.2(a) shows the interpolating polynomial in 31 Leja points for the scalar NLEP (4.2).

It is well known that rational interpolants $Q(\lambda)$ potentially exhibit faster convergence than polynomials, in particular, if the function to be approximated

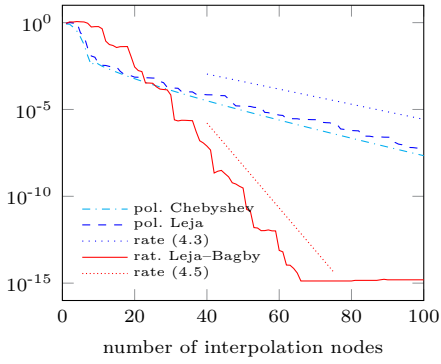


(a) Leja points

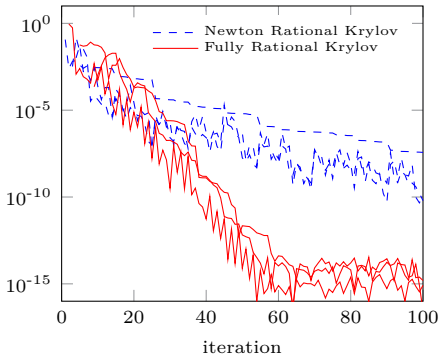


(b) Leja–Bagby points

Figure 4.2: Leja points versus Leja–Bagby points for $d = 30$.



(a) uniform interpolation error



(b) convergence of residuals

Figure 4.3: Scalar NLEP (4.2): polynomial interpolation in Leja points versus linear rational interpolation in Leja–Bagby pionts.

has singularities nearby Σ. In example (4.2), the *singularity set* Ξ of $A(\lambda)$ is the branch cut of the square root, $\Xi = (-\infty, 0]$. In this case it can be shown that there exist asymptotically optimal sequences of interpolation nodes σ_j in Σ and poles ξ_j in Ξ, so-called *Leja–Bagby points* [9], such that the resulting rational interpolants $Q(\lambda)$ converge considerably faster than (4.3), namely, like

$$\limsup_{d\to\infty} \big\|A(\lambda) - Q(\lambda)\big\|_{\Sigma}^{1/d} \leq \exp\left(-\frac{1}{\operatorname{cap}(\Sigma,\Xi)}\right) \lesssim \exp\left(-\frac{\pi^2}{\log(16\kappa)}\right), \quad (4.5)$$

with $\operatorname{cap}(\Sigma, \Xi)$ denoting a logarithmic capacity; see §4.2.3 for details.

Figure 4.2(b) shows the rational interpolant in 31 Leja–Bagby points for the scalar NLEP (4.2). Remark that, compared to polynomial interpolation in Figure 4.2(a), much more interpolation nodes are selected close to the branch cut of the square root. This results in a much faster decrease of the uniform error as shown in Figure 4.3(a). Note also that the convergence slopes of polynomial interpolation in Leja points and rational interpolation in Leja–Bagby points are very well predicted by the rates (4.3) and (4.5), respectively.

The convergence history of the root-finding iterations for the 3 real roots in Σ is given in Figure 4.3(b). In the Newton Rational Krylov method (Algorithm 3.2) we used polynomial interpolation in Leja points, while in the Fully Rational Krylov method (Algorithm 4.1) linear rational interpolation in Leja–Bagby points was used. This figure also illustrates that the accuracy of the eigenvalues is limited by the underlying polynomial or rational approximation. This explains the slower convergence of the Newton Rational Krylov method to the real roots of (4.1) in Σ.

4.2 Fully Rational Krylov method

The Fully Rational Krylov method [44] for solving nonlinear eigenvalue problems (4.1) uses a *rational Newton expansion* of the matrix-valued function $A(\lambda)$. This has the advantage that the interpolation nodes and poles can be added incrementally in a straightforward way by simply extending the companion-type linearization matrices, while the convergence of the approximation can be monitored by the magnitude of the computed rational divided differences.

The main difference between the variants of the Fully Rational Krylov method, which we will propose, lies in the way the construction of the linear rational approximation of $A(\lambda)$ is connected with the rational Krylov method. We distinguish 3 variants: static, dynamic, and hybrid.

Static variant In the spirit of a discretize-first approach, we construct in the static variant first the approximation and the corresponding linearization, in such a way that the approximation error is guaranteed to be uniformly small on the target set Σ. Choosing the interpolation nodes and poles based on arguments from potential theory, e.g., Leja–Bagby points, can lead to a fast uniform convergence of the approximation on the whole target set.

Next, the resulting generalized linear eigenvalue problem is solved by any method of choice, like the standard rational Krylov method. Note that the shifts in the rational Krylov method are chosen to solve this linear problem efficiently and are not necessarily related to the interpolation nodes of the linear rational approximation of $A(\lambda)$.

Dynamic variant In the spirit of the previous chapter, which is based on dynamic local approximation, the construction of the rational approximation and the application of the rational Krylov method can be tightly interwoven. By exploiting the structure of the linearization, a short starting vector, and the "trick" of choosing the shifts of the rational Krylov method identical to the interpolation nodes, the dynamic variant is a direct generalization of the Newton Rational Krylov method from polynomial to rational interpolation where the interpolation points can be chosen "on the fly". This method has the same computational cost per iteration as the Newton Rational Krylov method, but it may require considerably fewer iterations.

Even though the dynamic property of the algorithm is a major advantage, achieved by equating interpolation nodes and rational Krylov shifts, there might also be a price to pay for choosing the interpolation nodes and poles dynamically during the execution of the algorithm. An optimal choice in view of achieving a fast converging linearization, which typically means choosing interpolation nodes on the boundary of the target set, might not always be favorable for the rational Krylov method to converge quickly, e.g., if the eigenvalues are not located close to the boundary of the target set, and vice versa.

Hybrid variant Sometimes a combination of the two approaches is necessary, as we shall illustrate in Section 4.3. In the hybrid variant, we start with the dynamic variant where the shifts/interpolation nodes are determined by the underlying approximation problem, i.e., in such a way that we obtain a fast uniform convergence of the approximation.

At the moment the approximation has converged to sufficient accuracy, we freeze the linearization and continue the rational Krylov method on the fixed linearization matrices with the selection of shifts as for the standard rational Krylov method, i.e., not only on the boundary of the target set but also inside the target set.

Firstly, we show in §4.2.1 how $A(\lambda)$ can be approximated by linear rational interpolation and present the companion-type linearization of the resulting rational eigenvalue problem. Next, we describe in §4.2.2 only the *dynamic* variant in detail since the *static* variant is just a standard rational Krylov algorithm applied to the rational linearization pencil and the *hybrid* variant is just a combination of the two other ones. Finally, we discuss the choice of the algorithm parameters and the low rank exploitation of the coefficient matrices in §4.2.3 and §4.2.4, respectively.

4.2.1 Rational companion linearization

In order to solve the NLEP (4.1), we first approximate the matrix-valued function $A(\lambda)$ by a rational function $Q(\lambda)$ in a rational Newton basis. This results in a rational eigenvalue problem which can be linearized in a companion-type pencil.

Let the interpolation nodes and nonzero poles be given by $\sigma_0, \sigma_1, \ldots, \sigma_d$ and $\xi_1, \ldots, \xi_d$, respectively. We define the *rational basis functions*

$$b_i(\lambda) := \frac{1}{\beta_0} \prod_{k=1}^{i} \frac{\lambda - \sigma_{k-1}}{\beta_k(1 - \lambda/\xi_k)}, \qquad i = 0, 1, \ldots, d, \tag{4.6}$$

where $\beta_0, \beta_1, \ldots, \beta_d$ are nonzero scaling parameters. Note that its recurrence relation is given by

$$b_i(\lambda) = \frac{\lambda - \sigma_{i-1}}{\beta_i(1 - \lambda/\xi_i)} b_{i-1}(\lambda), \qquad i = 1, \ldots, d. \tag{4.7}$$

Then, the rational function

$$Q(\lambda) := \sum_{i=0}^{d} D_i b_i(\lambda), \tag{4.8}$$

with $D_i \in \mathbb{C}^{n\times n}$, interpolates $A(\lambda)$ in the interpolation nodes $\sigma_0, \sigma_1, \ldots, \sigma_d$. Note that the poles $\xi_1, \ldots, \xi_d$ are prescribed and we will assume that they are all distinct from the nodes $\sigma_0, \sigma_1, \ldots, \sigma_d$. In this case, $Q(\lambda)$ is a linear rational interpolant of type $[d, d]$ and hence guaranteed to exist uniquely. In particular, if $A(\lambda)$ itself is a rational eigenvalue problem of type $[d, d]$ with poles $\xi_1, \ldots, \xi_d$, then the interpolant $Q(\lambda) = A(\lambda)$ will be exact.

Note that if all poles ξ_i are at infinity, the functions $b_i(\lambda)$ reduce to the scaled Newton basis functions $n_i(\lambda)$, defined in (3.10), and the matrices D_i to the divided difference matrices A_i. Therefore, we will refer to D_j as *rational divided difference matrices*.

The NLEP (4.1) can now be approximated by the following rational eigenvalue problem

$$Q(\lambda)x = 0,$$

where $Q(\lambda)$ is defined by (4.8). Next, Theorem 2.13 can be generalized to linear rational interpolation, such that this REP can be linearized as follows

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x},$$

where

$$\mathbf{A} = \begin{bmatrix} D_0 & D_1 & \cdots & D_{d-2} & D_{d-1} - \sigma_{d-1}D_d/\beta_d \\ \sigma_0 I & \beta_1 I & & & \\ & \ddots & \ddots & & \\ & & \ddots & \beta_{d-2}I & \\ & & & \sigma_{d-2}I & \beta_{d-1}I \end{bmatrix}, \tag{4.9}$$

$$\mathbf{B} = \begin{bmatrix} D_0/\xi_d & D_1/\xi_d & \cdots & D_{d-2}/\xi_d & D_{d-1}\xi_d - D_d/\beta_d \\ I & \beta_1/\xi_1 I & & & \\ & \ddots & \ddots & & \\ & & \ddots & \beta_{d-2}/\xi_{d-2}I & \\ & & & I & \beta_{d-1}/\xi_{d-1}I \end{bmatrix}, \tag{4.10}$$

and

$$\mathbf{x} = \begin{bmatrix} b_0(\lambda)x \\ b_1(\lambda)x \\ \vdots \\ b_{d-1}(\lambda)x \end{bmatrix}. \tag{4.11}$$

Note that the last pole ξ_d plays a special role. In what follows it will be convenient to choose $\xi_d = \infty$. In this case the linearization has the same structure as in the Newton-type companion form used in the Newton Rational Krylov method (Section 3.2), and we can run exactly the same rational Krylov algorithm with a growing pencil.

Computing rational divided differences

As in the computation of divided differences, we can make use of matrix functions to compute the rational divided differences. Let the matrix-valued function $A(\lambda) \in \mathbb{C}^{n\times n}$ be written in the following form

$$A(\lambda) = \sum_{k=1}^{m} B_k f_k(\lambda),$$

where $B_k \in \mathbb{C}^{n\times n}$ are constant matrices, f_k are scalar functions of λ and $m \le n^2$. Then, the scalar functions f_k are approximated by rational Newton functions

$$f_k(\lambda) \approx \sum_{i=0}^{d} \delta_{ik} b_i(\lambda), \qquad k = 1, \ldots, m,$$

where δ_{ik} are the rational divided differences and b_i are the rational basis functions defined in (4.6). Consequently, the rational divided difference matrices D_i in (4.8) are linear combinations of the constant coefficient matrices B_k

$$D_i = \sum_{k=1}^{m} \delta_{ik} B_k, \qquad i = 0, 1, \ldots, d. \tag{4.12}$$

The scalar rational divided differences δ_{ij} can be computed by the following theorem which is inspired by the rational interpolation procedure underlying the pole and interpolation nodes (PAIN) method for matrix function approximation, see [42, Section 5.4.2] and [43]. In case all $\xi_j = \infty$, it reduces to Theorem 3.11.

Theorem 4.1. *Let the rational interpolant of type $[d, d]$ of a scalar function $f(\lambda)$ with prescribed interpolation nodes $\sigma_0, \sigma_1, \ldots, \sigma_d$ and poles $\xi_1, \ldots, \xi_d$ be given by*

$$q(\lambda) = \delta_0 b_0(\lambda) + \delta_1 b_1(\lambda) + \cdots + \delta_d b_d(\lambda),$$

where $b_i(\lambda)$ are the rational basis functions (4.6)*. Then, the rational divided differences $\delta_0, \ldots, \delta_d$ are the elements in the first row of $\beta_0 f(N^{-1}M)$, where*

$$M = \begin{bmatrix} \sigma_0 & \beta_1 & & & \\ & \sigma_1 & \beta_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & \beta_d \\ & & & & \sigma_d \end{bmatrix}, \qquad N = \begin{bmatrix} 1 & \beta_1/\xi_1 & & & \\ & 1 & \beta_2/\xi_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & \beta_d/\xi_d \\ & & & & 1 \end{bmatrix}.$$

Proof. See Güttel *et al.* [44, Theorem 3.1]. □

4.2.2 Dynamic variant

The *static* variant of the Newton Rational Krylov method is in fact the standard rational Krylov method applied to the rational linearization pencil (4.9)–(4.10) and where the shifts are free parameters. On the other hand, the *dynamic* variant is the generalization of the Newton Rational Krylov method to linear rational interpolation. In this case, the shifts are no free parameters and equal to the interpolation nodes. Finally, the *hybrid* variant is a combination of the two other ones. Therefore, we will only describe the *dynamic* variant in detail.

Building the rational Krylov subspace

In this paragraph, we generalize the results of §3.2.2 to linear rational interpolation. We start with the following key lemma.

Lemma 4.2. *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be defined by* (4.9)–(4.10) *with* $\xi_d = \infty$. *Suppose that only the first* $j+1$ *blocks of the vector* $\mathbf{y} \in \mathbb{C}^{dn}$ *are nonzero. Then for all* j, $0 \leq j \leq d-2$, *only the first* $j+1$ *blocks of the solution* $\mathbf{x}$ *of the linear system with shift* σ_j

$$(\mathbf{A} - \sigma_j \mathbf{B})\mathbf{x} = \mathbf{y}, \tag{4.13}$$

are nonzero. Furthermore, only the leading submatrices

$$\mathbf{A}_j = \begin{bmatrix} D_0 & D_1 & \dots & D_j \\ \sigma_0 I & \beta_1 I & & \\ & \ddots & \ddots & \\ & & \sigma_{j-1} I & \beta_j I \end{bmatrix}, \qquad \mathbf{B}_j = \begin{bmatrix} 0 & & & \\ I & \beta_1/\xi_1 I & & \\ & \ddots & \ddots & \\ & & I & \beta_j/\xi_j I \end{bmatrix},$$

of $\mathbf{A}$ *and* $\mathbf{B}$, *respectively, are needed to compute this nonzero part of* $\mathbf{x}$.

Proof. This lemma directly follows from the fact that the matrix $\mathbf{A} - \sigma_j \mathbf{B}$ in (4.13) is block triangular. □

The shifts in the Fully Rational Krylov method are, as in the Newton Rational Krylov method, not free parameters, but they are implicitly prescribed by the matrices $\mathbf{A}$ and $\mathbf{B}$ in (4.9)–(4.10), namely the nodes $\sigma_1, \sigma_2, \dots$. Furthermore, we suppose that only the first block component of the starting vector $\mathbf{v}_1 \in \mathbb{C}^{dn}$ is nonzero

$$\mathbf{v}_1 = \begin{bmatrix} v_1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}, \tag{4.14}$$

with $v_1 \in \mathbb{C}^n$. Consequently by Lemma 4.2, the rational Krylov subspace $\mathbf{V}$ has now also a growing block structure. This is summarized in the following lemma.

Lemma 4.3. *Suppose that the starting vector* $\mathbf{v}_1$ *of the rational Krylov method is of the form* (4.14) *and we use the nodes* $\sigma_1, \sigma_2, \ldots$ *as shifts. Then for all* j*,* $1 \leq j \leq d-1$*, only the first* j *blocks of the vector* $\mathbf{v}_j$*, generated by the rational Krylov method, are nonzero.*

Proof. The proof is similar to that of Lemma 3.6. □

An important consequence of Lemmas 4.2–4.3 is that, at each iteration j of the rational Krylov method, we only use the submatrices $\mathbf{A}_j$ and $\mathbf{B}_j$ for computing the next vector $\mathbf{v}_{j+1}$. Consequently, we only use the nodes $\sigma_0, \ldots, \sigma_j$ and poles $\xi_1, \ldots, \xi_j$, and the corresponding rational divided difference matrices $D_0, \ldots, D_j$.

Proposition 4.4. *The Ritz values* λ_i*, computed at iteration* j *of the rational Krylov method are independent of* d *as long as* $j < d$*. These Ritz values are also independent of* $\sigma_{j+1}, \ldots, \sigma_d$ *and* $\xi_{j+1}, \ldots, \xi_d$*.*

Proof. The proof is similar to that of Proposition 3.7. □

Corollary 4.5. *It is neither necessary to choose all nodes* σ_j *and poles* ξ_j *in advance, nor the degree* d *of the rational interpolant* $Q(\lambda)$*. Instead, in each iteration we can choose the next node and pole based on the results of the previous iterations. Therefore, the rational Krylov method can be implemented in an adaptive and an incremental way. The rational Krylov method is initialized with an interpolation node* σ_0 *and a particular starting vector, and can run until convergence by dynamically adding a node* σ_j *and pole* ξ_j *in each iteration.*

Algorithm

Based on Lemmas 4.2–4.3 and Corollary 4.5, the Fully Rational Krylov algorithm for solving the NLEP (4.1) can be dynamically implemented. Algorithm 4.1 gives an outline. As in the Newton Rational Krylov method, we can subdivide each iteration j into an expansion phase and a rational Krylov phase.

Firstly, in the expansion phase (lines 2–4), we choose at iteration j the next interpolation node σ_j, pole ξ_j, and scaling parameter β_j. Then, we compute the corresponding rational divided difference D_j in order to extend the linearization matrices $\mathbf{A}_{j-1}$ and $\mathbf{B}_{j-1}$ to $\mathbf{A}_j$ and $\mathbf{B}_j$, respectively. We also extend the matrix $\mathbf{V}_j$ with a zero block at the bottom.

Secondly, in the rational Krylov phase (lines 5–9), we perform a rational Krylov step with the extended matrices $\mathbf{A}_j$ and $\mathbf{B}_j$, and shift σ_j. In line 9, the Ritz values are computed in the same way as in the standard rational Krylov method (Algorithm 1.3).

Finally, in line 10, we take the first blocks of $\mathbf{x}_i$ as approximations for the nonlinear eigenvectors and check for convergence of the nonlinear eigenpairs $\left(\lambda_i, x_i^{[1]}\right)$.

Algorithm 4.1: Fully Rational Krylov method: dynamic variant [44]

Choose node σ_0, scaling parameter $\beta_0 = 1$, and starting vector $v_1 \in \mathbb{C}^n$.
for $j = 1, 2, \ldots$ **do**
  Expansion phase:
    Choose node, σ_j, pole ξ_j, and scaling parameter β_j.
    Compute rational divided difference matrix: D_j.
    Expand $\mathbf{A}_j$, $\mathbf{B}_j$ and $\mathbf{V}_j$.
  Rational Krylov phase:
    Set continuation combination: t_j.
    Compute $\widehat{\mathbf{v}} := (\mathbf{A}_j - \sigma_j \mathbf{B}_j)^{-1} \mathbf{B}_j \mathbf{V}_j t_j$.
    Orthogonalize: $\widetilde{\mathbf{v}} := \widehat{\mathbf{v}} - \mathbf{V}_j h_j$, where $h_j = \mathbf{V}_j^* \widehat{\mathbf{v}}$.
    Get new vector: $\mathbf{v}_{j+1} = \widetilde{\mathbf{v}} / h_{j+1,j}$, where $h_{j+1,j} = \|\widetilde{\mathbf{v}}\|$.
    Compute Ritzpairs: $(\lambda_i, \mathbf{x}_i)$.
    Nonlinear eigenpairs: $\left(\lambda_i, x_i^{[1]}\right)$ and test for convergence.
end

4.2.3 Choice of parameters

In each iteration j of Algorithm 4.1 we have the freedom to choose the interpolation nodes σ_j, poles ξ_j, and scaling parameters β_j. In this paragraph, we discuss how these parameters can automatically be chosen, in a numerically stable way, based on the target set Σ and the singularity set Ξ of $A(\lambda)$.

Interpolation nodes and poles

Choosing the parameters σ_j and ξ_j in a (near) optimal way is closely related to rational approximation problems on the target set Σ. These problems are in turn very closely related to logarithmic potential theory; see [67, 88] for introductions. For the purpose of this paper we only focus on linear rational interpolation with *prescribed* poles and nodes (as opposed to interpolation with *free* poles, a well-known special case of which is Padé approximation). This is a classical problem that has been studied extensively since the late 1960s by Bagby [9], Walsh [115, 117, 116], and others.

Let us assume that $\Sigma \subset \mathbb{C}$ is a simply connected compact set, and that $A(\lambda) = [a_{i,j}(\lambda)]$ is analytic in a simply connected open set $\Omega \supset \Sigma$. Let $Q(\lambda)$ be a rational interpolant of $A(\lambda)$ with interpolation nodes $\sigma_0, \sigma_1, \ldots, \sigma_d$ in Σ, and poles $\xi_1, \ldots, \xi_d$ outside Σ. Let each component of $Q(\lambda) = [q_{i,j}(\lambda)]$ have accuracy ε on Σ, i.e.,

$$\max_{i,j} \left\| a_{i,j}(\lambda) - q_{i,j}(\lambda) \right\|_\Sigma \leq \varepsilon.$$

Assume further that (λ^*, x) with $\lambda^* \in \Sigma$ and $\|x\|_2 = 1$ is an eigenpair for $Q(\lambda)$, i.e., $Q(\lambda^*)x = 0$. Then from

$$\big\|A(\lambda^*)x\big\|_2 = \big\|(A(\lambda^*) - Q(\lambda^*))x\big\|_2 \le \big\|A(\lambda^*) - Q(\lambda^*)\big\|_F \le n\varepsilon,$$

we find that a small component-wise uniform error of $Q(\lambda)$ for $A(\lambda)$ implies that (λ^*, x) has a small residual for the original NLEP (4.1). To obtain a reliable algorithm which attempts to find *all* eigenvalues in Σ, it is sensible to sample $A(\lambda)$ at nodes σ_j that guarantee a component-wise small error ε in $Q(\lambda)$ for all $\lambda \in \Sigma$.

In what follows we will drop the element indices from $a_{i,j}(\lambda) = a(\lambda)$ and $q_{i,j}(\lambda) = q(\lambda)$. The uniform interpolation error $\|a(\lambda) - q(\lambda)\|_\Sigma$ can be studied conveniently using the nodal rational functions

$$s_j(\lambda) = \frac{(\lambda - \sigma_0)(\lambda - \sigma_1)\cdots(\lambda - \sigma_j)}{(1 - \lambda/\xi_1)\cdots(1 - \lambda/\xi_j)}, \qquad j = 0, 1, \ldots.$$

Let Γ be a rectifiable closed curve in $\Omega \setminus \Sigma$, winding around Σ exactly once. Then, by the Walsh–Hermite integral representation of the interpolation error (see, e.g., [116, p. 50]) and standard estimation of integrals, we have for all $\lambda \in \Sigma$

$$|a(\lambda) - q(\lambda)| = \left|\frac{1}{2\pi i}\int_\Gamma \frac{s_d(\lambda)}{s_d(\zeta)}\,\frac{a(\zeta)}{(\zeta - \lambda)}d\zeta\right| \le C\frac{|s_d(\lambda)|}{\min_{\zeta\in\Gamma}|s_d(\zeta)|}, \tag{4.15}$$

for a constant C that only depends on Γ and $a(\lambda)$. The pair (Σ, Γ) is called a *condenser* [8, 40]. It can be shown [40, 66] that there exists a number $\mathrm{cap}(\Sigma, \Gamma) > 0$, called the *condenser capacity* of (Σ, Γ), such that

$$\limsup_{d\to\infty}\left(\frac{\max_{\lambda\in\Sigma}|s_d(\lambda)|}{\min_{\lambda\in\Gamma}|s_d(\lambda)|}\right)^{1/d} \ge \exp\left(-\frac{1}{\mathrm{cap}(\Sigma, \Gamma)}\right), \tag{4.16}$$

with equality if the points σ_j and ξ_j are distributed according to the so-called *signed equilibrium measure* on (Σ, Γ). A sequence of points that follow this distribution are the *Leja–Bagby points* for (Σ, Γ) [115, 9], which can be constructed as follows: start with an arbitrary $\sigma_0 \in \Sigma$, and then define the nodes $\sigma_j \in \Sigma$ and poles $\xi_j \in \Gamma$ recursively such that the following conditions are satisfied

$$\begin{aligned}\max_{\lambda\in\Sigma}|s_j(\lambda)| &= |s_j(\sigma_{j+1})|,\\ \inf_{\lambda\in\Gamma}|s_j(\lambda)| &= |s_j(\xi_{j+1})|,\end{aligned} \qquad j = 0, 1, \ldots.$$

By the maximum modulus principle for analytic functions, the points σ_j lie on $\partial\Sigma$, the boundary of Σ, and Γ can be replaced by its closed exterior,

say Ξ, without changing the capacity of $\operatorname{cap}(\Sigma, \Gamma) = \operatorname{cap}(\Sigma, \Xi)$. Combining the inequality (4.15) and (4.16) with equality, we arrive at the asymptotic convergence result

$$\limsup_{d\to\infty} \left\| A(\lambda) - Q(\lambda) \right\|_{\Sigma}^{1/d} \leq \exp\left(-\frac{1}{\operatorname{cap}(\Sigma, \Xi)} \right)$$

for linear rational interpolation at Leja–Bagby points. The convergence is thus geometric with a rate depending on the target set Σ and the poles on Ξ, which should stay away from Σ. In our numerical experiments we will typically use for Ξ a discretization of the singularity set of $f(\lambda)$. The determination of the numerical value $\operatorname{cap}(\Sigma, \Xi)$ is difficult for general condensers (Σ, Ξ). However, in some cases, including the example $(\Sigma, \Xi) = \big([\alpha, \beta], (-\infty, 0]\big)$ from Section 4.1, there are known closed formulas derived from conformal maps; see [43] for some examples, including the formula (4.5).

Scaling parameters

Once the points σ_j and ξ_j have been specified, it remains to choose appropriate scaling parameters β_j which can dramatically improve both the stability and the convergence of the Fully Rational Krylov method.

Note that, for a given fixed linear rational approximation $Q(\lambda)$ of $A(\lambda)$, changing a parameter β_j to $\alpha\beta_j$ has no influence other than scaling the divided difference matrices $D_j, \ldots, D_d$ to $\alpha D_j, \ldots, \alpha D_d$. Furthermore, scaling has no effect on the eigenvalues of the linearization pencil $(\mathbf{A}, \mathbf{B})$ given by (4.9)–(4.10), but the eigenvectors (4.11) will change since they depend on the scaling parameters β_j.

Although scaling has no effect on the eigenvalues, it definitely affects the converge of Algorithm 4.1 to these eigenvalues, since the scalar product, used in the orthogonalization process, and therefore also the constructed rational Krylov subspace change. Therefore, we choose the scaling prameters β_j in step 2 of Algorithm 4.1 such that all $b_j(\lambda)$ defined in (4.6) are of unit uniform norm on Σ, i.e., $\|b_j(\lambda)\|_{\Sigma} = 1$. It also guarantees that the evaluation of the functions $b_j(\lambda)$ during the sampling procedure is robust in the sense that it is not affected by numerical overflow or underflow.

This choice is motivated by the fact that for polynomial interpolation at Leja points the set Σ should be scaled to unit capacity for stability [81] or, alternatively, we can scale the Newton basis polynomials at each iteration. We are following the second approach, with the difference that we are now dealing with rational functions instead of polynomials. Our scaling also seems natural by inspecting the block components of the eigenvectors of the linearization in (4.11). If all $b_j(\lambda)$ have the same order of magnitude on Σ, then the eigenvectors

corresponding to eigenvalues λ in the target set Σ have evenly balanced block entries $b_j(\lambda)x$.

Under the condition that the poles ξ_j are away from Σ, the value $\|b_j(\lambda)\|_\Sigma$ is attained on the boundary $\Gamma = \partial\Sigma$. Therefore, we only need a sufficiently fine discretization $\Gamma_\nu = \{\gamma_1, \ldots, \gamma_\nu\}$ of Γ for a practically implementation and choose each β_j such that $\max_{\lambda\in\Gamma_\nu} |b_j(\lambda)| = 1$. In our numerical experiments we evaluated each $b_j(\lambda)$ at $\nu = 1000$ equispaced control points, a scalar computation which is of negligible constant cost when using the recursion (4.7).

The scaling such that $\|b_j(\lambda)\|_\Sigma = 1$ for all j is also convenient for error estimation: from a convergent expansion $Q_j(\lambda)$ of $A(\lambda)$ we find

$$\max_{\lambda\in\Sigma} \|A(\lambda) - Q_j(\lambda)\|_F \leq \|D_{j+1}\|_F + \|D_{j+2}\|_F + \cdots .$$

Thus, in the hybrid variant of the Fully Rational Krylov method the expansion of $Q_j(\lambda)$ can be stopped if this error is small enough. Since the divided differences D_j are computed as in (4.12), we use the scalar rational divided differences δ_{jk} for checking the accuracy of the expansion

$$\delta_j := \max_k |d_{jk}|.$$

This scalar quantity is readily available in the algorithm and can accurately be computed via matrix functions. For more details we refer to [44].

4.2.4 Low rank exploitation

In several applications the NLEP (4.1) consists of a polynomial part and a nonlinear part which is of low rank. See for example the `'gun'` problem and the `'particle in a canyon'` problem discussed in §1.1.2. We now generalize the low rank exploitation introduced in §3.2.4.

Suppose that the NLEP is defined as follows

$$A(\lambda)x = \left(\sum_{i=0}^{p} B_i \lambda^i + \sum_{i=1}^{m} C_i f_i(\lambda) \right) x = 0, \tag{4.17}$$

where $B_i, C_i \in \mathbb{C}^{n\times n}$ are constant matrices, $f_i(\lambda)$ are scalar functions of λ, $p \ll n^2$ and $m \ll n^2$. Furthermore, we assume that the matrices C_i have rank-revealing factorizations $C_i = L_i U_i^*$, where $L_i, U_i \in \mathbb{C}^{n\times r_i}$ are of full column rank $r_i \ll n$.

Approximating the scalar functions $f_i(\lambda)$ of (4.17) by linear rational interpolants with nodes $\sigma_0, \sigma_1, \ldots, \sigma_N$ and poles $\xi_1, \xi_2, \ldots, \xi_N$ yields

$$\widetilde{Q}(\lambda) = \sum_{i=0}^{d} \widetilde{D}_i b_i(\lambda) = \sum_{i=0}^{p} \left(\widetilde{B}_i + \widetilde{C}_i \right) b_i(\lambda) + \sum_{i=p+1}^{d} \widetilde{C}_i b_i(\lambda), \tag{4.18}$$

where $\widetilde{B}_i$ and $\widetilde{C}_i$ are defined as in (3.21).

Similarly as in §3.2.4, we obtain a companion-type reformulation where the pair $(\lambda, x \neq 0)$ is an eigenpair of the rational eigenvalue problem (4.18) if and only if

$$\widetilde{\mathbf{A}}\widetilde{\mathbf{x}} = \lambda \widetilde{\mathbf{B}}\widetilde{\mathbf{x}},$$

where

$$\widetilde{\mathbf{A}} = \begin{bmatrix} \widetilde{D}_0 & \widetilde{D}_1 & \cdots & \widetilde{D}_p & \widetilde{L}_{p+1} & \cdots & \widetilde{L}_{d-2} & \widetilde{L}_{d-1} - \sigma_{d-1}\widetilde{L}_d/\beta_d \\ \sigma_0 I & \beta_1 I & & & & & & \\ & \ddots & \ddots & & & & & \\ & & \sigma_{p-1} I & \beta_p I & & & & \\ & & & \sigma_p \widetilde{U}^* & \beta_{p+1} I & & & \\ & & & & \sigma_{p+1} I & \beta_{p+2} I & & \\ & & & & & \ddots & \ddots & \\ & & & & & & \sigma_{d-2} I & \beta_{d-1} I \end{bmatrix}$$

$$\widetilde{\mathbf{B}} = \begin{bmatrix} \widetilde{D}_0/\xi_d & \widetilde{D}_1/\xi_d & \cdots & \widetilde{D}_p/\xi_d & \widetilde{L}_{p+1}/\xi_d & \cdots & \widetilde{L}_{d-2}/\xi_d & \widetilde{L}_{d-1}/\xi_d - \widetilde{L}_d/\beta_d \\ I & \gamma_1 I & & & & & & \\ & \ddots & \ddots & & & & & \\ & & I & \gamma_p I & & & & \\ & & & \widetilde{U}^* & \gamma_{p+1} I & & & \\ & & & & I & \gamma_{p+2} I & & \\ & & & & & \ddots & \ddots & \\ & & & & & & I & \gamma_{d-1} I \end{bmatrix}$$

with $\widetilde{D}_i = \widetilde{B}_i + \widetilde{C}_i$ and $\gamma_i = \beta_i/\xi_i$, and

$$\widetilde{\mathbf{x}} = \begin{bmatrix} b_0(\lambda)x \\ b_1(\lambda)x \\ \vdots \\ b_p(\lambda)x \\ b_{p+1}(\lambda)\widetilde{U}^* x \\ \vdots \\ b_{d-2}(\lambda)\widetilde{U}^* x \\ b_{d-1}(\lambda)\widetilde{U}^* x \end{bmatrix}.$$

For this type of linearization we can also prove Lemmas 4.2–4.3.

4.3 Numerical experiments

In Section 4.1 we demonstrated the Fully Rational Krylov method as a root finder for a scalar problem and illustrated the differences between polynomial and rational interpolation. Here, we apply Algorithm 4.1 to two large-scale applications. Firstly, we consider again the `'gun'` problem for which we compare polynomial versus rational interpolation and dynamic versus static interpolation. Secondly, we solve the `'particle in a canyon'` problem with the dynamic and static version of the Fully Rational Krylov method.

All numerical experiments are performed in MATLAB version 7.14.0 (R2012a) on a Dell Latitude notebook running an Intel(R) Core(TM) i5-2540M CPU @ 2.60GHz quad core processor with 8 GB RAM. Our experiments can be reproduced with the publicly available code of [44].

Before presenting the results of the numerical experiments, we first describe 4 variants of Algorithm 4.1 to be compared below.

Variant P: a *dynamic* polynomial version of Algorithm 4.1, which is in fact the Newton Rational Krylov method introduced in [106]. In this variant, $A(\lambda)$ is approximated by interpolating polynomials $P_j(\lambda)$ with cyclically repeated interpolation nodes $\sigma_j \in \Omega_{\text{cycl}} \subset \Sigma$ and all poles $\xi_j = \infty$. The shifts of the rational Krylov space are chosen equal to the interpolation nodes σ_j in order to make the algorithm dynamic.

Variant R: a *dynamic* rational version of Algorithm 4.1, whereby $A(\lambda)$ is approximated by linear rational interpolants $Q_j(\lambda)$ with cyclically repeated interpolation nodes $\sigma_j \in \Omega_{\text{cycl}}$ and poles $\xi_j \in \Xi$ selected in Leja–Bagby style. Again, the shifts are chosen equal to the interpolation nodes σ_j.

Variant H: a *hybrid* rational version of Algorithm 4.1. As long as the linear rational interpolant $Q_j(\lambda)$ has not converged yet, we choose Leja–Bagby interpolation nodes $\sigma_j \in \Sigma$ and poles $\xi_j \in \Xi$. Upon convergence we truncate and freeze the rational expansion $Q_j(\lambda)$. Next, we switch to cyclically repeated interpolation nodes $\sigma_j \in \Omega_{\text{cycl}}$ in order to obtain faster convergence of the rational Krylov method to eigenvalues located in the interior of Σ. The poles ξ_j are still selected in Leja–Bagby style. The shifts of the rational Krylov space are again chosen equal to the interpolation nodes σ_j in order to make the algorithm dynamic.

Variant S: a *static* rational version of Algorithm 4.1. We first determine the linear rational approximation $Q(\lambda)$ such that $Q(\lambda) \approx A(\lambda)$ for all $\lambda \in \Sigma$. For the computation of $Q(\lambda)$ we select Leja–Bagby interpolation nodes $\sigma_j \in \Sigma$ and poles $\xi_j \in \Xi$, and use the same truncation criterion as for *Variant H*. Then, once the linearization is fixed, we use the rational Krylov method with shifts in Ω_{cycl} for solving the generalized eigenvalue problem in a nondynamic way.

4.3.1 Gun problem

We reconsider the 'gun' problem, see §1.1.2, which is a large-scale problem that models a radio-frequency gun cavity and is of the form

$$A(\lambda)x = \Big(K - \lambda M + i\sqrt{\lambda - \sigma_1^2}\, W_1 + i\sqrt{\lambda - \sigma_2^2}\, W_2\Big)x = 0, \tag{4.19}$$

where M, K, W_1 and W_2 are real symmetric matrices of size 9956×9956, K is positive semidefinite and M is positive definite. As in [14], we take $\sigma_1 = 0$ and $\sigma_2 = 108.8774$. The complex square root $\sqrt{\cdot}$ corresponds to the principal branch. For measuring the convergence of an approximate eigenpair (λ, x), we used the relative residual norm [68]

$$E(\lambda, x) = \frac{\|A(\lambda)x\|_2/\|x\|_2}{\|K\|_1 + |\lambda|\|M\|_1 + \sqrt{|\lambda - \sigma_1^2|}\|W_1\|_1 + \sqrt{|\lambda - \sigma_2^2|}\|W_2\|_1}.$$

The target set Σ is the upper half disk with centre 250^2 and radius $300^2 - 200^2$, see Figure 4.4(a). The singularity set $\Xi = (-\infty, \sigma_2^2]$ corresponds to the union of branch cuts of the square roots. In Algorithm 4.1 we have discretized Ξ by logarithmically spaced points $(\sigma_2^2 - 10^{-8+16j/10^4})$ with $j = 0, 1, \ldots, 10^4$.

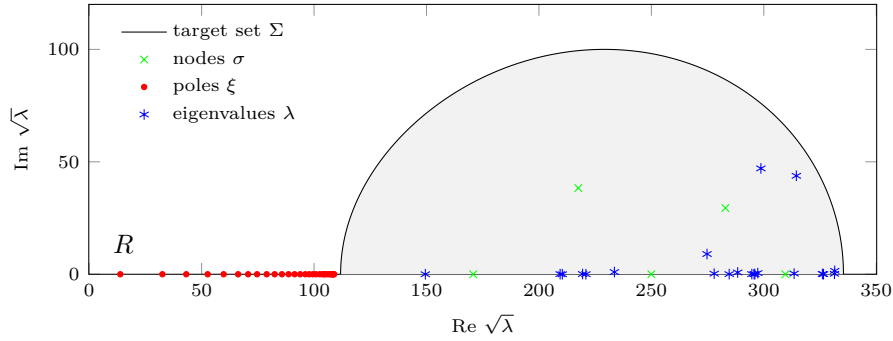
Thanks to the automatic scaling strategy described in §4.2.3, we can solve the NLEP (4.19) directly with Algorithm 4.1, instead of first transforming Σ to roughly the upper half of the unit disk as in §3.3.2. We have also exploited the low-rank structure of the coefficient matrices W_1 and W_2 as explained in §4.2.4.

Polynomial (P) versus rational (R) interpolation

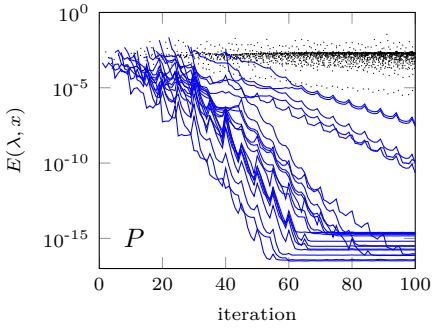
In a first experiment we compare *Variant P* and *Variant R*. In both variants, we chose 5 cyclically repeated interpolation nodes in Σ, indicated by "$\times$" in Figure 4.4(a). The corresponding Leja–Bagby poles, selected in *Variant R*, are indicated by "$\bullet$".

The convergence history of the eigenpairs computed with *Variant P* and *Variant R* are given in Figure 4.4(b) and Figure 4.4(c), respectively. Note that in these figures the solid and dotted lines correspond to eigenvalues lying inside and outside the target set Σ, respectively. From these figures, we can see that the eigenvalues computed with *Variant R* converge much faster than these computed with *Variant P* and this with similar total computation cost. Hence, we conclude that the Fully Rational Krylov method can be significantly faster than the Newton Rational Krylov method.

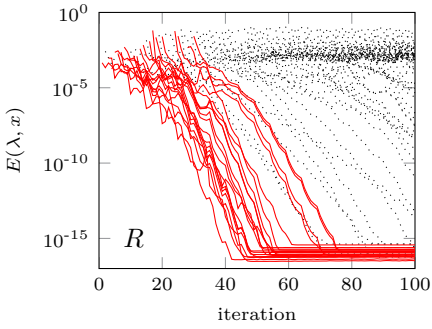
We will now look at the corresponding convergence of the approximations of $A(\lambda)$ by polynomial interpolants $P_j(\lambda)$ and rational interpolants $Q_j(\lambda)$. Figure 4.5 shows the maxima of the scalar rational divided differences in every



(a) *Variant R*: nodes, poles and eigenvalues



(b) *Variant P*: convergence of eigenpairs



(c) *Variant R*: convergence of eigenpairs

Figure 4.4: Results for the 'gun' problem: (a) Approximate eigenvalues for the original NLEP (4.19) obtained with Variant R, (b) convergence history for Variant P, and (c) convergence history for Variant R.

iteration j of Algorithm 4.1. We see that in *Variant P* (solid line) there is no convergence for $j \to \infty$. Thus, it is possible to miss some eigenvalues since in this case the underlying linearized polynomial eigenvalue problem does not approximate the original NLEP accurately in the whole target set Σ. See also Table 4.1 below, which shows that only 17 of 21 eigenvalues are found with *Variant P*.

On the other hand, in *Variant R* (dashed line), the approximations $Q_j(\lambda)$ of $A(\lambda)$ converges slowly as j increases. In order to get faster convergence of the rational approximations of $A(\lambda)$, we can use Leja–Bagby interpolation nodes and poles. In Figure 4.5 we see that the error of the rational interpolants with Leja–Bagby points in *Variant H* (dotted line) decreases much faster than the one of the rational interpolants with cyclically repeated nodes in *Variant R* (dashed line).

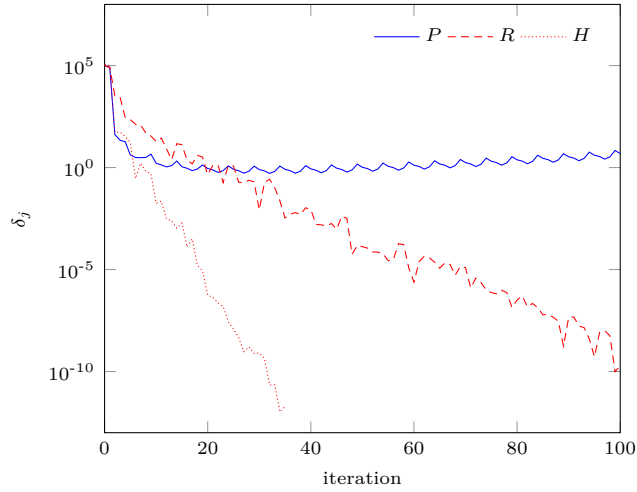


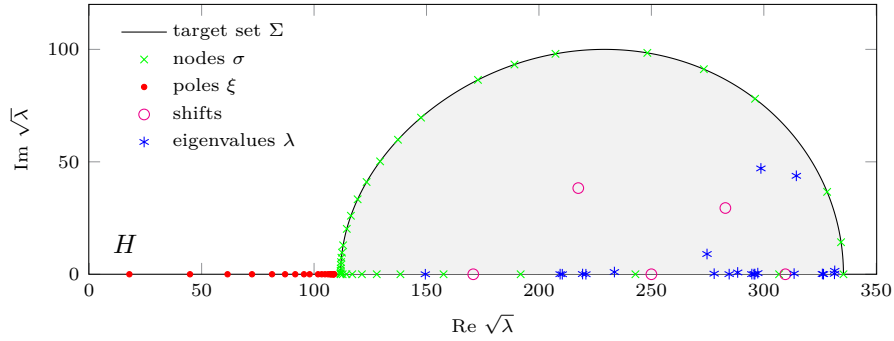
Figure 4.5: Convergence of the approximations of $A(\lambda)$ for the '`gun`' problem: Variant P (solid line), Variant R (dashed line), and Variant H (dotted line).

Hybrid (H) versus static (S) variant

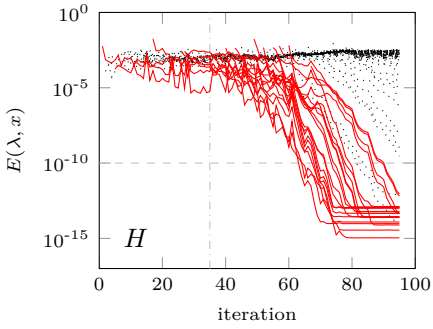
In this experiment we compare *Variant H* and *Variant S*. In both variants we chose Leja–Bagby interpolation nodes and poles, indicated in Figure 4.6(a) by "$\times$" and "$\bullet$", respectively. We also set the tolerance for the relative residual norm $E(\lambda)$ to $\texttt{tol} = 10^{-10}$.

Although Leja–Bagby points are near-optimal for uniform convergence of the rational expansions $Q_j(\lambda)$, their use as shifts of the rational Krylov space may not be advantageous for quickly finding eigenvalues inside Σ. Hence, upon convergence of $Q_j(\lambda)$, we recommend to either switch to interpolation nodes in the interior of Σ and apply the truncation strategy explained in §4.2.3, or to use the static variant.

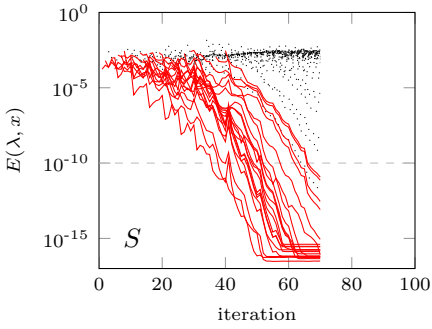
In *Variant H* we used Leja–Bagby points only for the first $j = 35$ iterations, until the approximations $Q_j(\lambda)$ have converged. Next, we applied the truncation strategy and froze the linearization. As a result the rational Krylov vectors do not grow any more, resulting in lower memory consumption and a cheaper orthogonalization process. We then switched to cyclically repeated interpolation nodes, indicated by "$\circ$" in Figure 4.6(a), in order to obtain faster convergence to eigenvalues located in the interior of Σ. The resulting convergence history is shown in Figure 4.6(b). In this figure we see that during the expansion phase, i.e., when the shifts of the rational Krylov space are still chosen on $\partial\Sigma$, there is very slow convergence for some eigenvalues close to $\partial\Sigma$ and where the density of interpolation nodes is high. From iteration $j = 36$ onwards, that is when



(a) *Variant H*: nodes, poles, shifts and eigenvalues



(b) *Variant H*: convergence of eigenpairs



(c) *Variant S*: convergence of eigenpairs

Figure 4.6: Results for the ‘gun’ problem: (a) Approximate eigenvalues for the original NLEP (4.19) obtained with Variant H, (b) convergence history for Variant H, and (c) convergence history for Variant S.

we use cyclically repeated interpolation nodes in the interior of Σ, we notice in Figure 4.6(b) a fast and very regular convergence for all eigenvalues in Σ.

During the expansion phase of *Variant H* we again observe slow convergence for some eigenvalues. As before this is because the shifts of the rational Krylov space are chosen equal to the Leja–Bagby interpolation nodes in order to make the algorithm dynamic. Therefore, in *Variant S* we first determine the rational approximation $Q(\lambda)$ and then freeze the linearization. Next, the generalized eigenvalue problem is solved with the standard rational Krylov method with cyclically repeated shifts, indicated by “∘” in Figure 4.6(a). The corresponding convergence history of the eigenpairs computed with *Variant S* is given in Figure 4.6(c). Note that *Variant S* only requires 70 iterations compared to 95 in *Variant H* for computing all eigenvalue inside Σ.

Timings and memory usage

We now compare timings and memory usage of the 4 different variants of Algorithm 4.1 for solving the 'gun' problem. In all experiments, we used the reorthogonalization strategy of [62] and exploited the low-rank structure of the nonlinear part of (4.19) as explained in §4.2.4. Hence, the orthogonalization cost is not dominant compared to the one for the system solves. In case of cyclically repeated shifts $\sigma_j \in \Omega_{\text{cycl}}$ of the rational Krylov space, we reused the LU factors of $A(\sigma_j)$.

A comparison for solving the 'gun' problem with *Variant P*, *Variant R*, *Variant H*, and *Variant S* is given in Table 4.1, where the memory usage also includes the storage of the LU factors (~ 75 MB each). From this table, we see that the computational cost and memory usage of *Variant P* and *Variant R* are very similar: both variants require the same number of system solves and LU factorizations, and the rational Krylov vectors have the same length. There only is a difference in computation cost of the orthogonalization process, due to different amount of reorthogonalization: only 35% of the iterations of *Variant P* require reorthogonalization, whereas in *Variant R* reorthogonalization is required in 89% of the iterations.

Variant H requires more computation time, due to the higher number of LU factorizations, but in this variant the approximations of $A(\lambda)$ converge. Therefore *Variant H* is more robust than *Variant P* and *Variant R*. The slightly lower memory usage in *Variant H* is due to the stopping of this method after 95 iterations and because the rational Krylov vectors do not grow any more after the linearization is frozen.

Table 4.1 also shows that *Variant S* is the most efficient variant for solving the 'gun' problem. Firstly, this variant requires only 70 iterations, since the rational Krylov process only starts after the approximation has converged. Together with the reuse of LU factors, this results in a low system solving cost. Secondly, although the rational Krylov vectors are of the same length as in *Variant H*, the memory usage in *Variant S* is lower since less iterations are needed to compute all the eigenvalues in Σ. On the other hand, compared to the other variants, *Variant S* does no longer have the property of being dynamic.

Table 4.1: Timings and memory usage for the 'gun' problem.

Method	# It.	# λ	Sys. solves	Orthog.	Total CPU	Memory
Variant P	100	17	7.1 s	2.1 s	9.6 s	$\sim$ 460 MB
Variant R	100	21	7.1 s	2.9 s	10.7 s	$\sim$ 460 MB
Variant H	95	21	26.5 s	2.7 s	30.2 s	$\sim$ 449 MB
Variant S	70	21	6.0 s	1.1 s	8.1 s	$\sim$ 435 MB

4.3.2 Particle in a canyon problem

We consider the `'particle in a canyon'` problem, see §1.1.2, which models the Schrödinger equation for a particle in a potential well attached to a number of contacts.

In this example, the particle has mass $0.2\,\mathrm{m_e}$ and the two-dimensional potential has a canyon-like shape with a canyon length, width and depth of 2.2 nm, 4 nm and 3 eV, respectively, while the width and depth of the valley in the contacts is 2 nm and 3 eV, respectively. The Schrödinger equation is discretized on a $4 \times 10\,\mathrm{nm}^2$ grid. The corresponding nonlinear eigenvalue problem is

$$A(\lambda)x = \left(H - \lambda I - \textstyle\sum_{k=1}^{n_z} e^{i\sqrt{m(\lambda-\alpha_k)}} L_k U_k^*\right)x = 0, \tag{4.20}$$

where $H \in \mathbb{R}^{16281\times 16281}$ is symmetric, $L_k, U_k \in \mathbb{R}^{16281\times 2}$, $m = 0.2$ and $n_z = 81$. The branch points are defined by $\alpha_k \in \mathbb{R}$ and sorted in ascending order.

We take the interval between the first and second branch point as target set $\Sigma = [\alpha_1 + \varepsilon, \alpha_2 - \varepsilon]$, with $\alpha_1 \approx -0.198$, $\alpha_2 \approx -0.132$ and $\varepsilon = 10^{-4}$. In order to make Σ branch cut free, we define the branch cut corresponding to the first nonlinear term in (4.20) as $(-\infty, \alpha_1]$, whereas all other branch cuts are defined as $[\alpha_k, +\infty)$ for $k = 2, 3, \ldots, n_z$. The singularity set $\Xi = (-\infty, \alpha_1] \cup [\alpha_2, +\infty)$ is the union of all branch cuts. We have discretized Ξ by the union of 10^4 logarithmically spaced points on $[-10^6, \alpha_1]$ and $[\alpha_2, 10^6]$. In all experiments we used again the reorthogonalization strategy of [62] and exploited the low-rank structure of the nonlinear part of (4.20) as explained in §4.2.4.

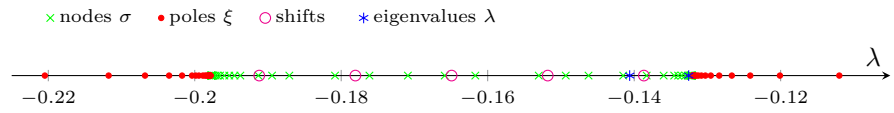
Another approach for solving the `'particle in a canyon'` problem was proposed in Vandenberghe *et al.* [109]. In this case, holomorphic extensions are used to get rid of the branch cuts. For more details, we refer to Chapter 7.

Hybrid (H) versus static (S) variant

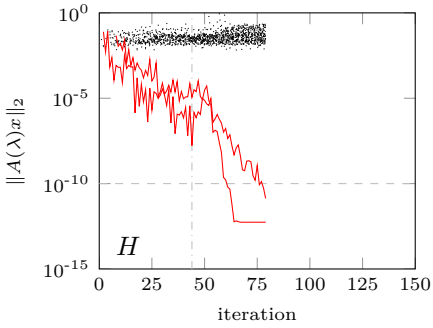
For the NLEP (4.20) we only compare *Variant H* and *Variant S* of Algorithm 4.1. The Leja–Bagby interpolation nodes and poles, used in both experiments, are indicated in Figure 4.7(a) by "×" and "•", respectively. The cyclically repeated shifts are indicated by "○" in Figure 4.7(a). We also set the tolerance for the residual norm $\|A(\lambda)x\|_2$ to $\texttt{tol} = 10^{-10}$. The convergence histories of the eigenpairs computed with *Variant H* and *Variant S* are given in Figure 4.7(b) and Figure 4.7(c), respectively.

Timings and memory usage

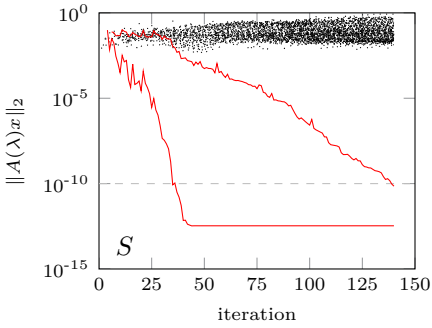
We compare timings and memory usage of *Variant H* and *Variant S* for solving the `'particle in a canyon'` problem.



(a) *Variant H*: nodes, poles, shifts and eigenvalues



(b) *Variant H*: convergence of eigenpairs



(c) *Variant S*: convergence of eigenpairs

Figure 4.7: Results for the '`particle in a canyon`' problem: (a) approximate eigenvalues for the NLEP (4.20) obtained with Variant H, (b) convergence history for Variant H, and (c) convergence history for Variant S.

A comparison of the computation time and memory usage is shown in Table 4.2, where the memory usage also includes the storage of the LU factors ($\sim$ 22 MB each). Table 4.2 shows that *Variant H* is the most efficient, both in terms of computation time and memory usage. In contrast to the previous example, the Leja–Bagby nodes in this example lie *inside* the region of interest Σ (an interval) and hence are good choices for the shifts of the rational Krylov space. As a consequence, the Ritz pairs start converging already during the expansion phase of the algorithm, see Figure 4.7(b), resulting in almost half the number of iterations required by *Variant S*. Note also that due to the larger number of iterations in *Variant S*, the orthogonalization process becomes the dominant computational cost.

Table 4.2: Timings and memory usage for the '`particle in a canyon`' problem.

Method	# It.	# λ	Sys. solves	Orthog.	Total CPU	Memory
Variant H	79	2	12.2 s	3.5 s	18.5 s	$\sim$ 234 MB
Variant S	140	2	6.5 s	11.8 s	21.8 s	$\sim$ 286 MB

4.4 Conclusions

In this chapter, we introduced the Fully Rational Krylov method for solving the nonlinear eigenvalue problem. We considered three different variants which all use a new linearization based on linear rational interpolation.

Firstly, the *dynamic* variant is a generalization of the Newton Rational Krylov method where the linearization is efficiently intertwined with the rational Krylov process. In case all poles ξ_j are infinite, we obtain exactly the Newton Rational Krylov method. However, the Fully Rational Krylov method has an automatic scaling strategy, via an estimation of the logarithmic capacity of (Σ, Ξ) using control points on the boundary of Σ, which can significantly improve both the stability and the convergence of the method. Furthermore, this automatic scaling strategy has the advantage that a transformation of the original NLEP, such that the region of interest is approximately the unit interval or unit circle, is not necessary anymore.

Secondly, the *hybrid* variant starts with using near-optimal Leja–Bagby points in order to obtain a fast and uniform convergence on the target set of the rational approximation of $A(\lambda)$. However, these Leja–Bagby points may not always be advantageous for the rational Krylov iteration to quickly find the targeted eigenvalues of the linearization. Due to its fast convergence, the rational expansion can be truncated after some iterations, which ultimately allows us to freely choose the shifts of the rational Krylov space. Therefore, after truncation we switch to shifts σ_j in the interior of Σ in order to obtain a fast rational Krylov convergence.

Thirdly, the *static* variant constructs, before starting the rational Krylov process, a rational approximation of $A(\lambda)$ and corresponding linearization in such a way that the approximation error is guaranteed to be uniformly small on the target set. Next, the resulting generalized linear eigenvalue problem is solved by the standard rational Krylov method. The difference with [98] is that the static variant of the Fully Rational Krylov method does not require low rank nonlinear terms and can adopt polynomials of high degree as illustrated in the numerical examples. In addition, all variants of the Fully Rational Krylov method can also greatly benefit from low rank nonlinear terms.

The numerical experiments have revealed that these variants of the Fully Rational Krylov method are viable. We found that for all examples, the dynamic variant largely outperforms the Newton Rational Krylov method, both in speed and reliability. We expect the hybrid and static variant to be useful for applications. For problems with eigenvalues on an interval nearby singularities, as in the `'particle in a canyon'` problem, the hybrid version appears to be the most efficient and reliable method in terms of number of iterations. For problems with eigenvalues on a two-dimensional target set, as in the `'gun'` problem, the static version appears to be both fast and reliable.

Chapter 5

Operator Setting

The operator setting leads to another approach for solving the nonlinear eigenvalue problem (NLEP)

$$A(\lambda)x = 0, \tag{5.1}$$

where $\lambda \in \Omega \subset \mathbb{C}$, $x \in \mathbb{C}^n \setminus \{0\}$, and $A(\lambda)$ is analytic on Ω. Here, we define a linear operator $\mathcal{A}$ whose eigenvalues are the solutions of the NLEP. In general this linear operator can be infinite-dimensional.

In order to compute the eigenvalues of the linear operator, we make use of the Arnoldi method in a function setting. With a particular choice of the starting function and a particular choice of the scalar product, the structure of the operator can be exploited such that, although the operator is in general infinite-dimensional, only standard linear algebra operations on vectors and matrices of finite size are involved.

This chapter is organized as follows. Firstly, we introduce the operator $\mathcal{A}$ and show its equivalence with the NLEP in Section 5.1. Next, the inverse of the operator $\mathcal{A}$ is used in a function setting of Arnoldi's method in Section 5.2 resulting in the Infinite Arnoldi method. In Section 5.3 we illustrate its connection with a discretize-first approach. Finally, we summarize the main results in Section 5.5.

5.1 Operator reformulation

The key point for solving a NLEP via the operator setting is that (5.1) can equivalently be expressed with the following linear operator eigenvalue problem

$$\mathcal{A}\varphi = \lambda\varphi, \tag{5.2}$$

where $\mathcal{A}$ is in general an infinite dimensional linear operator and the eigenfunctions φ are defined as $\varphi : \mathbb{C} \to \mathbb{C}^n \setminus \{0\}$.

Since we assume that the matrix-valued function $A(\lambda)$ in (5.1) is analytic, we can use the concise functional analysis notation $A(\frac{d}{d\theta})$ defined by the Taylor expansion

$$A\left(\frac{d}{d\theta}\right) := \sum_{i=0}^{\infty} \frac{A^{(i)}(0)}{i!} \frac{d^i}{d\theta^i}, \tag{5.3}$$

where $A^{(i)}$ denotes the ith derivative of A.

Firstly, we define the operator $\mathcal{A}$ and prove the equivalence with the NLEP in §5.1.1. Secondly, we derive the operator $\mathcal{A}^{-1}$ and discuss its properties in §5.1.2.

5.1.1 Operator equivalence

We define the operator $\mathcal{A}$ and its domain $\mathcal{D}(\mathcal{A})$ as follows.

Definition 5.1 (Operator $\mathcal{A}$)**.** *Let $X := C_\infty(\mathbb{R}, \mathbb{C}^n)$ be the set of infinitely differential functions $\varphi : \mathbb{R} \to \mathbb{C}^n$ and denote the function variable θ. We define the linear operator $\mathcal{A} : \mathcal{D}(\mathcal{A}) \subseteq X \to X$ and its domain $\mathcal{D}(\mathcal{A})$ as*

$$\mathcal{D}(\mathcal{A}) := \left\{ \varphi \in X : A\left(\frac{d}{d\theta}\right)\varphi(0) = 0 \right\}, \tag{5.4}$$

$$(\mathcal{A}\varphi)(\theta) := \frac{d\varphi}{d\theta}(\theta), \qquad \varphi \in \mathcal{D}(\mathcal{A}), \tag{5.5}$$

where A is given in (5.1)*.*

The connection between the eigenvalues and eigenvectors of the nonlinear eigenvalue problem (5.1) and the eigenvalues and eigenfunctions of the linear operator $\mathcal{A}$ is given by the following theorem.

Theorem 5.2. *Let the operator $\mathcal{A}$ be defined by Definition 5.1.*

1. *If the pair (λ, x) is an eigenpair of the NLEP* (5.1)*, then the pair $(\lambda, xe^{\lambda\theta})$ is an eigenpair of the operator eigenvalue problem* (5.2)*.*

2. *If the pair (λ, φ) is an eigenpair of the operator eigenvalue problem* (5.2)*, then there exists a vector x such that $\varphi(\theta) = xe^{\lambda\theta}$ and the pair (λ, x) is an eigenpair of the NLEP* (5.1)*.*

Proof. The proof of part *1* immediately follows from the definition of $\mathcal{A}$. Let φ be given by $\varphi(\theta) = xe^{\lambda\theta}$. Then, $\varphi \in \mathcal{D}(\mathcal{A})$ since

$$\left(A\left(\frac{d}{d\theta}\right)xe^{\lambda\theta}\right)(0) = A(\lambda)x = 0.$$

Hence,

$$\mathcal{A}\varphi = \varphi' = \lambda xe^{\lambda\theta} = \lambda\varphi.$$

For the proof of part *2* we first show that an eigenfunction of $\mathcal{A}$ is always exponential in θ. By definition, the solution of the differential equation

$$\varphi' = \mathcal{A}\varphi = \lambda\varphi,$$

is always of the form $\varphi(\theta) = ce^{\lambda\theta}$. Using (5.4) and choosing $x = \varphi(0)$ completes the proof. □

5.1.2 Inverse operator

In the next section, we will use the Arnoldi method in a function setting to compute eigenvalues of (5.2). Therefore, we also define the operator $\mathcal{A}^{-1}$ in order to compute eigenvalues with small magnitude.

Definition 5.3 (Operator $\mathcal{A}^{-1}$)**.** *The inverse of the linear operator $\mathcal{A}$, given by Definition 5.1, exists iff $A(0)$ is nonsingular and is given by*

$$\mathcal{D}(\mathcal{A}^{-1}) := \left\{\varphi \in X : \left\|\left(A\left(\frac{d}{d\theta}\right)\int_0^\theta \varphi(s)ds\right)(0)\right\| < \infty\right\}, \tag{5.6}$$

$$(\mathcal{A}^{-1}\varphi)(\theta) := \int_0^\theta \varphi(s)ds + C_\varphi, \qquad \varphi \in \mathcal{D}(\mathcal{A}^{-1}), \tag{5.7}$$

where the integration constant $C_\varphi \in \mathbb{C}^n$ is given by

$$C_\varphi = -A(0)^{-1}\left(A\left(\frac{d}{d\theta}\right)\int_0^\theta \varphi(s)ds\right)(0).$$

The inverse of the operator $\mathcal{A}$ is also a linear operator, which is formally stated by the following lemma.

Lemma 5.4 (Linearity)**.** *The operator $\mathcal{A}^{-1}$ satisfies the linearity property*

$$\mathcal{A}^{-1}(\alpha\varphi + \beta\psi) = \alpha\mathcal{A}^{-1}\varphi + \beta\mathcal{A}^{-1}\psi,$$

for any two functions $\varphi, \psi \in \mathcal{D}(\mathcal{A}^{-1})$ and any two constant scalars $\alpha, \beta \in \mathbb{C}$.

Proof. See Jarlebring *et al.* [52, Proposition 2]. □

5.2 Infinite Arnoldi method

The Infinite Arnoldi method for solving the nonlinear eigenvalue problem (5.1) was introduced by Jarlebring *et al.* [52, 51] and relies on the following 2 main ingredients:

1. the reformulation of the NLEP (5.1) as an operator eigenvalue problem (5.2) with the same eigenvalues;

2. solving the operator eigenvalue problem by Arnoldi's method with the linear operator $\mathcal{A}^{-1}$, a constant starting function, and a particular scalar product.

Since applying the operator $\mathcal{A}^{-1}$ to a polynomial results again in a polynomial, the Krylov subspace generated during the Arnoldi process is a vector space of polynomials. By representing these polynomials in a *degree-graded basis* and by a *suitable choice of the scalar product*, the structure of the infinite dimensional linear operator $\mathcal{A}^{-1}$ can be exploited such that the method only involves linear algebra operations applied to matrices of finite size.

Firstly, we introduce in §5.2.1 the Arnoldi method in a function setting. Next, we discuss in §5.2.2 how to exploit the structure of the map $\mathcal{A}^{-1}$ where the functions are represented by polynomial in a degree-graded basis and introduce the Infinite Arnoldi algorithm in §5.2.3.

5.2.1 Arnoldi method in a function setting

Similar to the linear eigenvalue problem, the Arnoldi method can be applied to the operator $\mathcal{A}^{-1}$, instead of to a matrix, for computing the smallest eigenvalues of the operator eigenvalue problem (5.2). An outline is given in Algorithm 5.1.

Algorithm 5.1: Arnoldi method for $\mathcal{A}^{-1}$

Choose function $\phi_1 \in \mathcal{D}(\mathcal{A}^{-1})$, such that $\langle \phi_1, \phi_1 \rangle = 1$.
for $j = 1, 2, \ldots$ **do**
  Compute: $\psi := \mathcal{A}^{-1}\phi_j$.
  for $i = 1, \ldots, j$ **do**
    Orthogonalize: $\psi := \psi - h_{i,j}\phi_i$, where $h_{i,j} = \langle \psi, \phi_i \rangle$.
  end
  Get new function: $\phi_{j+1} = \psi / h_{j+1,j}$, where $h_{j+1,j} = \sqrt{\langle \psi, \psi \rangle}$.
  Compute Ritzpairs: (λ_i, φ_i) and test for convergence.
end

The Arnoldi method applied to the operator $\mathcal{A}^{-1}$ builds an orthogonal basis of the Krylov subspace

$$\mathcal{K}_j(\mathcal{A}^{-1}, \phi_1) := \operatorname{span}\left\{\phi_1, \mathcal{A}^{-1}\phi_1, \ldots, \mathcal{A}^{-(j-1)}\phi_1\right\},$$

where $\phi_1 \in \mathcal{D}(\mathcal{A}^{-1}) \subset C_\infty(\mathbb{R}, \mathbb{C}^n)$. Simultaneously, an orthogonal projection of $\mathcal{A}^{-1}$ onto this subspace is constructed by using a Gram–Schmidt orthogonalization process associated with an arbitrary scalar product $\langle \cdot\,, \cdot \rangle$. The Ritz pairs are computed from the Hessenberg matrix in the same way as in the standard shift-and-invert Arnoldi method (Algorithm 1.2).

5.2.2 Building the Krylov subspace

In order to compute approximations for the eigenvalues with small magnitude of the operator eigenvalue problem (5.2), we will use the inverse operator $\mathcal{A}^{-1}$ instead of $\mathcal{A}$. Note that the starting function ϕ_1 and the scalar product are still free parameters we can choose.

In case we start the Arnoldi method for $\mathcal{A}^{-1}$ with a constant starting function ϕ_1, the iterates ϕ_j will be all polynomials of degree $j-1$. Furthermore, if we represent these functions by polynomials in a degree-grade basis, the coefficient map of the infinite dimensional linear operator $\mathcal{A}^{-1}$ involves in every iteration j of Algorithm 5.1 only $j+1$ vectors of length n.

Coefficient map

Suppose that we represent the functions $\phi_1, \phi_2, \ldots$ in Algorithm 5.1 by polynomials in a degree-graded basis. We define the integration map as follows.

Definition 5.5 (Integration map)**.** *Let $\{p_i\}_{i=0}^{\infty}$ be a sequence of degree-graded polynomials such that p_i is of degree i and has a non-zero leading coefficient and let $p_0(\theta) = 1$. Then, the integration map of the polynomials $p_0, \ldots, p_{d-1}$ is defined as follows*

$$\begin{bmatrix} p_0(\theta) \\ p_1(\theta) \\ \vdots \\ p_{d-1}(\theta) \end{bmatrix} = L_d \begin{bmatrix} p_1'(\theta) \\ p_2'(\theta) \\ \vdots \\ p_d'(\theta) \end{bmatrix},$$

where $L_d \in \mathbb{R}^{d\times d}$ for any $d \in \mathbb{N}$.

Then, the action $\mathcal{A}^{-1}$ on polynomials given in a degree-grade basis results in the following coefficient map.

Theorem 5.6 (Coefficient map [52])**.** *Let the polynomials $p_0, \dots, p_d$ and the integration map L_d be defined by Definition 5.5. Moreover, let ϕ be given by*

$$\phi(\theta) := \sum_{i=0}^{d-1} p_i(\theta) v_i, \tag{5.8}$$

where $v_0, \dots, v_{d-1} \in \mathbb{C}^n$ are the vector coefficients in the polynomial basis $p_0, \dots, p_{d-1}$. Then, the coefficients of $\psi := \mathcal{A}^{-1}\phi$, i.e.,

$$\psi(\theta) = (\mathcal{A}^{-1}\phi)(\theta) =: \sum_{i=0}^{d} p_i(\theta) w_i,$$

where $w_0, w_1, \dots, w_d \in \mathbb{C}^n$ are given by

$$\begin{bmatrix} w_1 & \cdots & w_d \end{bmatrix} = \begin{bmatrix} v_0 & \cdots & v_{d-1} \end{bmatrix} L_d, \tag{5.9}$$

and

$$w_0 = C_\phi - \sum_{i=1}^{d} p_i(0) w_i. \tag{5.10}$$

Proof. From the expansion of ϕ in (5.8) and by using the integration map defined in Definition 5.5, we have

$$\begin{aligned} \int_0^\theta \phi(s) ds &= \int_0^\theta \begin{bmatrix} v_0 & \cdots & v_{d-1} \end{bmatrix} \begin{bmatrix} p_0(s) \\ \vdots \\ p_{d-1}(s) \end{bmatrix} ds, \\ &= \begin{bmatrix} v_0 & \cdots & v_{d-1} \end{bmatrix} L_d \begin{bmatrix} p_1(\theta) - p_1(0) \\ \vdots \\ p_d(\theta) - p_d(0) \end{bmatrix}. \end{aligned} \tag{5.11}$$

Next, inserting (5.11) into (5.7) yields

$$\begin{aligned} \psi(\theta) &= \int_0^\theta \phi(s) ds + C_\phi, \\ &= \begin{bmatrix} v_0 & \cdots & v_{d-1} \end{bmatrix} L_d \begin{bmatrix} p_1(\theta) \\ \vdots \\ p_d(\theta) \end{bmatrix} + \left(C_\phi - \begin{bmatrix} v_0 & \cdots & v_{d-1} \end{bmatrix} L_d \begin{bmatrix} p_1(0) \\ \vdots \\ p_d(0) \end{bmatrix} \right), \end{aligned}$$

and using $p_0(\theta) := 1$ results in (5.9)–(5.10). This completes the proof. □

Thus, by choosing a constant function $\phi_1(\theta) = v_1$ with $v_1 \in \mathbb{C}^n \setminus \{0\}$ as starting function in Algorithm 5.1, the functions ϕ_j have the following form

$$\phi_j(\theta) = \sum_{i=0}^{j-1} p_i(\theta) v_j^{[i+1]},$$

where the vector coefficients $v_j^{[i+1]} \in \mathbb{C}^n$. In case we stack $v_j^{[1]}, \dots, v_j^{[j]}$ on top of each other, the Krylov subspace $\mathcal{K}_j(\mathcal{A}^{-1}, \phi_1)$, which is a vector space of polynomials of degree $j-1$, can be represented by the matrix

$$\mathbf{V}_j = \begin{bmatrix} v_1^{[1]} & v_2^{[1]} & \cdots & v_j^{[1]} \\ & v_2^{[2]} & \cdots & v_j^{[2]} \\ & & \ddots & \vdots \\ & & & v_j^{[j]} \end{bmatrix}.$$

Hence, we obtain a dynamically growing Krylov subspace with the same block structure as in the dynamic polynomial and rational interpolation methods of Chapters 3 and 4.

Scalar product

In case we represent ϕ with vector coefficients in a degree-graded basis, the action of $\mathcal{A}^{-1}\phi$ can be expressed by only using matrices and vectors of size n. We now propose a scalar product which can also be implemented using only vectors of size n.

Definition 5.7 (Scalar product [52])**.** *Let the polynomials $p_0, \dots, p_d$ be defined by Definition 5.5. Then, we define the scalar product*

$$\langle \phi, \psi \rangle := \sum_{i=0}^{d} w_i^* v_i, \tag{5.12}$$

where the functions ϕ and ψ are given by

$$\phi(\theta) = \sum_{i=0}^{d} p_i(\theta) v_i, \qquad \psi(\theta) = \sum_{i=0}^{d} p_i(\theta) w_i,$$

with $v_i, w_i \in \mathbb{C}^n$ for $i = 0, 1, \dots, d$.

To prove that the map (5.12) defines indeed a scalar product directly follows from the verification of the properties of a scalar product. Note that the scalar product, defined in Definition 5.7, corresponds to summing up the Euclidian scalar products of the vector coefficients in the polynomial basis in which the functions are represented.

5.2.3 Algorithm

By using the coefficient map for $\mathcal{A}^{-1}$ in a degree-graded basis together with the corresponding scalar product defined in Definition 5.7, Algorithm 5.1 can be implemented such that only standard linear algebra operations on vectors and matrices of finite size are involved. Moreover, the Infinite Arnoldi method, outlined in Algorithm 5.2[1], for solving the NLEP (5.1) can dynamically be implemented based on Theorem 5.6. Therefore, we can subdivide each iteration into two main phases: an expansion phase followed by an Arnoldi phase.

Algorithm 5.2: Infinite Arnoldi method [52]

1 Choose starting vector $v_1 \in \mathbb{C}^n$.
for $j = 1, 2, \ldots$ **do**
  EXPANSION PHASE:
2 Expand: L_j and $\mathbf{V}_j$.
  ARNOLDI PHASE:
3 Compute $\widehat{\mathbf{v}}$:

$$\begin{bmatrix} \widehat{v}^{[2]} & \cdots & \widehat{v}^{[j+1]} \end{bmatrix} = \begin{bmatrix} v_j^{[1]} & \cdots & v_j^{[j]} \end{bmatrix} L_j, \tag{a}$$

$$\widehat{v}^{[1]} = C_\phi - \sum_{i=1}^{j} p_i(0)\widehat{v}^{[i+1]}. \tag{b}$$

4 Orthogonalize: $\widetilde{\mathbf{v}} := \widehat{\mathbf{v}} - \mathbf{V}_j h_j$, where $h_j = \mathbf{V}_j^* \widehat{\mathbf{v}}$.
5 Get new vector: $\mathbf{v}_{j+1} = \widetilde{\mathbf{v}}/h_{j+1,j}$, where $h_{j+1,j} = \|\widetilde{\mathbf{v}}\|$.
6 Compute Ritzpairs: (λ_i, φ_i).
7 Nonlinear eigenpairs: $\left(\lambda_i, x_i^{[1]}\right)$ and test for convergence.
end

Firstly, in the expansion phase (line 2), we extend the lower triangular integration map matrix with one row at the bottom and one column at the right. We also extend the matrix $\mathbf{V}_j$ with a zero block at the bottom.

Secondly, in the Arnoldi phase (lines 3–6), we perform an Arnoldi step with the operator $\mathcal{A}^{-1}$. Although the operator is in general infinite dimensional, only matrix-vector operations of finite dimension are involved in this case. In line 6, the Ritz values λ_i are computed as the reciprocal eigenvalues of H_j, i.e., the upper $j \times j$ part of the Hessenberg matrix $\underline{H}_j$ obtained from the

[1] Remark that in [52] the formulas for the coefficient map are expressed in terms of a reformulation of (5.1), i.e., $\lambda B(\lambda)x = x$.

orthogonalization process. The Ritz functions φ_i are obtained as follows

$$\varphi_i(\theta) = \sum_{k=0}^{j} p_k(\theta) x_i^{[k+1]}, \qquad i = 1, \ldots, j,$$

where

$$\mathbf{x}_i = \mathbf{V}_{j+1} \underline{H}_j s_i.$$

Note that for converged Ritz functions the vectors $\mathbf{x}_i$ have the following Kronecker structure $\mathbf{x}_i = a \otimes x_i^{[1]}$, where $a \in \mathbb{C}^j$.

Finally, in line 7, we take the first blocks of $\mathbf{x}_i$ as approximations for the nonlinear eigenvectors and check for convergence of the nonlinear eigenpairs $\left(\lambda_i, x_i^{[1]}\right)$.

5.2.4 Low rank exploitation

In several applications the NLEP (4.1) consists of a polynomial part and a nonlinear part which is of low rank. See for example the `'gun'` problem and the `'particle in a canyon'` problem discussed in §1.1.2. We now propose, inspired by §3.2.4, a low rank exploitation for the Infinite Arnoldi method.

Suppose that the NLEP is defined as follows

$$A(\lambda)x = \left(\sum_{i=0}^{p} B_i \lambda^i + \sum_{i=1}^{m} C_i f_i(\lambda) \right) x = 0, \tag{5.13}$$

where $B_i, C_i \in \mathbb{C}^{n \times n}$ are constant matrices, $f_i(\lambda)$ are scalar functions of λ, $p \ll n^2$ and $m \ll n^2$. Furthermore, we assume that the matrices C_i of low rank $r_i \ll n$. Note that the definition of $A(\lambda)$ in (5.13) corresponds to low rank higher order terms in the Taylor expansion (5.3)

$$A^{(i)}(0) = L_i U^*, \qquad i \geq p + 1, \tag{5.14}$$

where $U \in \mathbb{C}^{n \times r}$ with $r \leq r_1 + \cdots + r_m$ is a matrix with orthonormal columns and $L_i \in \mathbb{C}^{n \times r}$ for $i \geq p + 1$. We define the following operator.

Definition 5.8 (Operator $\mathcal{F}$)**.** *Let $X := C_\infty(\mathbb{R}, \mathbb{C}^n)$ be the set of infinitely differential functions $\varphi : \mathbb{R} \to \mathbb{C}^n$ and denote the function variable θ. We define the low rank operator $\mathcal{F} : X \to X$ as*

$$(\mathcal{F}\varphi)(\theta) := \sum_{i=0}^{p} I_n \varphi^{(i)}(0) \frac{\theta^i}{i!} + \sum_{i=p+1}^{\infty} U U^* \varphi^{(i)}(0) \frac{\theta^i}{i!},$$

where U is given in (5.14)*.*

Using the low rank operator $\mathcal{F}$, we can reformulate the nonlinear eigenvalue problem (5.13) as the following operator eigenvalue problem

$$\left(\mathcal{F}\mathcal{A}^{-1}\right)\varphi = \frac{1}{\lambda}\varphi, \tag{5.15}$$

where the operator $\mathcal{A}$ is defined in Definition 5.1 and the eigenfunctions φ are defined as $\varphi : \mathbb{C} \to \mathbb{C}^n \setminus \{0\}$. The eigenvalue equivalence between (5.13) and (5.15) is given by the following theorem.

Theorem 5.9. *Let the operators $\mathcal{A}$ and $\mathcal{F}$ be defined by Definition 5.1 and Definition 5.8, respectively.*

1. *If the pair (λ, x) is an eigenpair of the NLEP (5.1), then the pair (λ, φ), where*
$$\varphi(\theta) = \sum_{i=0}^{p} x\frac{(\lambda\theta)^i}{i!} + \sum_{i=p+1}^{\infty} UU^*x\frac{(\lambda\theta)^i}{i!},$$
is an eigenpair of the operator eigenvalue problem (5.15).

2. *If the pair (λ, φ) is an eigenpair of the operator eigenvalue problem (5.15), then there exists a vector $x = \varphi(0)$ such that the pair (λ, x) is an eigenpair of the NLEP (5.1).*

Proof. See Van Beeumen *et al.* [105, Theorem 2.1]. □

Solving the NLEP with low rank nonlinear part (5.13) by the Infinite Arnoldi method applied to $\mathcal{F}\mathcal{A}^{-1}$ yields a reduction of both the computation time required for the orthogonalization and the memory required to store the matrix $\mathbf{V}$.

By choosing a constant function $\phi_1(\theta) = v_1$ with $v_1 \in \mathbb{C}^n \setminus \{0\}$ as starting function in Algorithm 5.1 and due to the inclusion of $\mathcal{F}$, the functions ϕ_j have the following form

$$\phi_j(\theta) := \sum_{i=0}^{p} p_i(\theta)v_j^{[i+1]} + \sum_{i=p+1}^{j-1} p_i(\theta)U\widehat{v}_j^{[i+1]}, \tag{5.16}$$

where the vector coefficients $v_j^{[i+1]} \in \mathbb{C}^n$ and $\widehat{v}_j^{[i+1]} \in \mathbb{C}^r$. Consequently, the coefficient map of Theorem 5.6 can be adapted for the compressed representation of $\varphi_j(\theta)$ (5.16). For more details, we refer to [105].

By using the coupling of basis and scalar product, we can carry out the scalar product of two functions directly in the compressed representation, which is summarized by the following theorem [105].

Theorem 5.10. *Let the polynomials $p_0, \dots, p_d$ be defined by Definition 5.5 and consider the following two functions*

$$\phi(\theta) = \sum_{i=0}^{p} p_i(\theta) v_i + \sum_{i=p+1}^{d} p_i(\theta) U \widehat{v}_i,$$

$$\psi(\theta) = \sum_{i=0}^{p} p_i(\theta) w_i + \sum_{i=p+1}^{d} p_i(\theta) U \widehat{w}_i,$$

where $v_i, w_i \in \mathbb{C}^n$ for $i = 0, 1, \dots, p$, $\widehat{v}_i, \widehat{w}_i \in \mathbb{C}^r$ for $i = p+1, \dots, d$, and the matrix U has orthonormal columns. Then, the scalar product (5.12) *reduces to*

$$\langle \phi, \psi \rangle := \sum_{i=0}^{p} w_i^* v_i + \sum_{i=p+1}^{d} \widehat{w}_i^* \widehat{v}_i.$$

5.3 Connection with discretize-first approach

The Infinite Arnoldi method applied to a linear infinite-dimension operator reformulation of the NLEP (5.1), started with a constant starting function and equipped with a particular product, only involves linear algebra operations with finite matrices. Therefore, it is also possible to characterize Algorithm 5.2 in a different way and show its connection with a discretize-first approach.

Firstly, we show in §5.3.1 the equivalence between the Infinite Arnoldi method and the standard Arnoldi method applied to a large companion-type pencil. More precisely, we illustrate that the action $\mathcal{A}^{-1}$ on functions represented in a degree-graded basis is equivalent to the action of a finite matrix. Together with the scalar product definition (5.12), we can show that performing j iterations of the Infinite Arnoldi method yields in exact arithmetic identical Hessenberg matrices as j iterations of the standard Arnoldi method applied to a finite pencil of dimension dn with $d > j$. Hence, both methods result in the same approximation for the eigenvalues of the NLEP.

Next, we consider in §5.3.2 the monomial basis for representing the functions in the Infinite Arnoldi method. In this case Algorithm 5.2 is identical to the Taylor–Arnoldi method (Algorithm 3.1) where a truncated Taylor series is used to obtain the linearization pencil.

Finally, we confine to delay eigenvalue problems in §5.3.3. We show that the Delay Arnoldi method [49], which is the Infinite Arnoldi method applied to the inverse delay operator with a shifted and scaled Chebyshev scalar product, is equivalent to the standard Arnoldi method applied to a companion-type reformulation of the spectral discretization of the delay operator.

5.3.1 Interpretation as Arnoldi's method on matrices

We will illustrate that j iterations of the Infinite Arnoldi method, outlined in Algorithm 5.2, are equivalent to j iterations of the standard Arnoldi method applied to a finite pencil of dimension dn with $d > j$.

We start by rewriting step 3 of Algorithm 5.2 as the solution of the following linear system

$$\left(\begin{bmatrix} 1 & p_1(0) & \cdots & p_j(0) \\ 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \end{bmatrix} \otimes I_n\right) \begin{bmatrix} \widehat{v}^{[1]} \\ \widehat{v}^{[2]} \\ \vdots \\ \widehat{v}^{[j+1]} \end{bmatrix} = \begin{bmatrix} C_0 & \cdots & C_{j-1} \\ & L_j^T \otimes I_n & \end{bmatrix} \begin{bmatrix} v_j^{[1]} \\ \vdots \\ v_j^{[j]} \end{bmatrix},$$

where $C_i \in \mathbb{C}^{n\times n}$ are given by

$$C_i := -A(0)^{-1}\left(A\left(\frac{d}{d\theta}\right)\int_0^\theta p_i(s)ds\right)(0), \qquad i = 0, 1, \ldots.$$

Let $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{dn\times dn}$ be defined as follows

$$\mathbf{A} := \left[\begin{array}{c|ccc} 1 & p_1(0) & \cdots & p_{d-1}(0) \\ \hline 0 & & I_{d-1} & \end{array}\right] \otimes I_n, \tag{5.17}$$

$$\mathbf{B} := \left[\begin{array}{ccccc} C_0 & C_1 & \cdots & C_{d-1} \\ \hline & & \underline{L}_d^T \otimes I_n & \end{array}\right], \tag{5.18}$$

where $\underline{L}_d \in \mathbb{R}^{d\times(d-1)}$ is the leading part of the integration matrix L_d. Then, $\widehat{\mathbf{v}}$ in step 3 of Algorithm 5.2 can also be obtained as follows

$$\widehat{\mathbf{v}} = \mathbf{A}_j^{-1}\mathbf{B}_j\underline{\mathbf{v}}_j, \tag{5.19}$$

where $\underline{\mathbf{v}}_j \in \mathbb{C}^{(j+1)n}$ is the extended version of $\mathbf{v}_j$ with a zero block at the bottom and $\mathbf{A}_j$ and $\mathbf{B}_j$ are the leading $(j+1)\times(j+1)$ submatrices of (5.17) and (5.18), respectively.

Hence, performing j iterations of Algorithm 5.2, where we replace step 3 with (5.19), is equivalent to j iterations of the standard Arnoldi method applied to the GEP

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x},$$

for any $d > j$ and with starting vector

$$\mathbf{v}_1 = \begin{bmatrix} v_1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}. \tag{5.20}$$

This is elaborated in what follows. To be more specific, the action of the infinite-dimensional operator $\mathcal{A}^{-1}$ is equivalent to the action of the finite matrix $\mathbf{A}^{-1}\mathbf{B}$ in the following sense.

Lemma 5.11 (Action equivalence)**.** *Let the polynomials $p_0, \dots, p_d$ with $d > j$ be defined by Definition 5.5. Moreover, let $v_i \in \mathbb{C}^n$ for $i = 0, 1, \dots, j-1$ and $w_i \in \mathbb{C}^n$ for $i = 0, 1, \dots, j$ be the vector coefficients of*

$$\phi(\theta) = \sum_{i=0}^{j-1} p_i(\theta) v_i,$$

$$\psi(\theta) = \sum_{i=0}^{j} p_i(\theta) w_i,$$

such that these coefficients fulfill

$$\begin{bmatrix} w_0 \\ \vdots \\ w_{j-1} \\ w_j \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \mathbf{A}^{-1}\mathbf{B} \begin{bmatrix} v_0 \\ \vdots \\ v_{j-1} \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

where $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{dn \times dn}$ are defined by (5.17)–(5.18)*. Then, the operator $\mathcal{A}^{-1}$ corresponding to the nonlinear eigenvalue problem* (5.1) *is equivalent to $\mathbf{A}^{-1}\mathbf{B}$ in the sense that*

$$\psi = \mathcal{A}^{-1}\phi.$$

Proof. See Jarlebring *et al.* [52, Lemma 7]. □

Based on the equivalence between the action of $\mathcal{A}^{-1}$ and the action of $\mathbf{A}^{-1}\mathbf{B}$ in Lemma 5.11 and the definition of the scalar product in Definition 5.7, we are now able to formulate the equivalence between the Infinite Arnoldi method and the standard Arnoldi method applied to the pencil $\mathbf{A} - \lambda\mathbf{B}$.

Theorem 5.12 (Hessenberg equivalence)**.** *Let $\mathbf{A}$ and $\mathbf{B}$ be defined by* (5.17) *and* (5.18)*, respectively. Then, the Hessenberg matrix $\underline{H}_j$ as well as the matrix of basis vectors $\mathbf{V}_{j+1}$ obtained by j iterations, with $j < d$, of the standard Arnoldi method applied to $\mathbf{A}^{-1}\mathbf{B}$ with starting vector $\mathbf{v}_1$* (5.20) *are in exact arithmetic identical to the ones obtained after j iterations of the Infinite Arnoldi method applied to $\mathcal{A}^{-1}$ with the scalar product defined in Definition 5.7 and the starting vector v_1.*

Proof. See Jarlebring *et al.* [52, Theorem 8]. □

5.3.2 Truncate Taylor series

In case we use the monomial basis to represent the functions

$$\phi_j(\theta) = \sum_{i=0}^{j-1} \theta^i v_j^{[i+1]}, \qquad j = 1, 2, \ldots,$$

the integration map L_d in Definition 5.5 is the diagonal matrix

$$L_d = \begin{bmatrix} 1 & & & \\ & \frac{1}{2} & & \\ & & \ddots & \\ & & & \frac{1}{d} \end{bmatrix}.$$

Furthermore, $p_i(0) := 0$ for all $i > 0$ and

$$C_i = -A(0)^{-1} \left(A\left(\frac{d}{d\theta}\right) \int_0^\theta s^i ds \right)(0) = -A(0)^{-1} A^{(i+1)}(0),$$

for $i = 0, 1, \ldots$. Therefore, the matrices (5.17)–(5.18) are $\mathbf{A} = I_{dn}$ and

$$\mathbf{B} = \begin{bmatrix} -A(0)^{-1} & \\ & I_{(d-1)n} \end{bmatrix} \begin{bmatrix} A^{(1)}(0) & A^{(2)}(0) & \cdots & A^{(d-1)}(0) & A^{(d)}(0) \\ I_n & & & & \\ & \frac{1}{2} I_n & & & \\ & & \ddots & & \\ & & & \frac{1}{d-1} I_n & 0 \end{bmatrix},$$

respectively. Hence,

$$\mathbf{A}^{-1}\mathbf{B} = \begin{bmatrix} S_1 & S_2 & \cdots & S_{d-1} & S_d \\ I & 0 & & & \\ & \frac{1}{2} I & 0 & & \\ & & \ddots & \ddots & \\ & & & \frac{1}{d-1} I & 0 \end{bmatrix},$$

where $S_i = -A(0)^{-1} A^{(i)}(0)$ for $i = 1, \ldots, d$ is exactly the same matrix as the one formed by the matrices (3.7)–(3.8) with $\sigma = 0$, $\alpha_i = i!$, $\beta_i = \frac{1}{i}$, and $A_i = A^{(i)}(0)$. Consequently, we find that the Infinite Arnoldi method applied to the operator $\mathcal{A}^{-1}$ in the monomial basis and with the corresponding scalar product (5.12) is equivalent to the Taylor–Arnoldi method (see Section 3.1).

5.3.3 Spectral discretization of delay operator

Consider the delay eigenvalue problem (DEP):

$$A(\lambda)x := \left(\lambda I - A_0 - \sum_{i=1}^{m} A_i e^{-\lambda\tau_i}\right) x = 0, \tag{5.21}$$

where $A_0, A_1, \ldots, A_m \in \mathbb{C}^{n\times n}$ and $\tau_1, \ldots, \tau_m \in \mathbb{R}$ are the discrete delays. Without loss of generality, we order the delays and let $0 =: \tau_0 < \tau_1 < \cdots < \tau_m$. The delay eigenvalue problem (5.21) is a special case of the general nonlinear eigenvalue problem (5.1). Consequently, we can also define an equivalent infinite dimensional linear operator $\mathcal{A}$.

Definition 5.13 (Delay operator $\mathcal{A}$)**.** *Let $X := C([-\tau_m, 0], \mathbb{C}^n)$ be the Banach space of continuous functions $\varphi : [-\tau_m, 0] \to \mathbb{C}^n$, equipped with the supremum norm, and denote the function variable θ. Then, we define the linear operator $\mathcal{A} : \mathcal{D}(\mathcal{A}) \subseteq X \to X$ and its domain $\mathcal{D}(\mathcal{A})$ as*

$$\mathcal{D}(\mathcal{A}) := \left\{ \varphi \in X : \frac{d\varphi}{d\theta} \in X,\ \frac{d\varphi}{d\theta}(0) = A_0\varphi(0) + \sum_{i=1}^{m} A_i\varphi(-\tau_i) \right\},$$

$$(\mathcal{A}\varphi)(\theta) := \frac{d\varphi}{d\theta}(\theta), \qquad \theta \in [-\tau_m, 0],$$

where $A_0, A_1, \ldots, A_m$ are given in (5.21)*.*

We now show that the Delay Arnoldi method [49], i.e., the Infinite Arnoldi method applied to the inverse delay operator $\mathcal{A}^{-1}$ with a shifted and scaled Chebyshev scalar product, can also be interpreted as the standard Arnoldi method applied to a pencil obtained by a spectral discretization of the delay operator defined in Definition 5.13.

Spectral discretization

A common approach to compute the spectrum of an infinite-dimensional delay operator $\mathcal{A}$ is to approximate it by a matrix $\mathbf{M}$ obtained from a spectral discretization method, see, e.g., [102, 19]. Next, the eigenvalues of $\mathbf{M}$ are computed with an eigenvalue solver of choice.

Consider a mesh Ω of d distinct points in the interval $[-\tau_m, 0]$

$$\Omega = \{-\tau_m \le \theta_1 < \theta_2 < \cdots < \theta_{d-1} < \theta_d = 0\}.$$

Now consider the discretized problem where we have replaced X with the space X_d of discrete functions defined over the mesh Ω, i.e., any function $\varphi \in X$ is

discretized into a block vector $\mathbf{z} \in X_d$

$$\mathbf{z} := \begin{bmatrix} z_1 \\ \vdots \\ z_d \end{bmatrix} \in \mathbb{C}^{dn},$$

where $z_i := \varphi(\theta_i) \in \mathbb{C}^n$ for $i = 1, \ldots, d$.

Let $\mathcal{P}$ be the unique $\mathbb{C}^n$ valued interpolating polynomial of degree smaller than or equal to $d-1$, satisfying

$$\mathcal{P}(\theta_i) = z_i, \qquad i = 1, \ldots, d.$$

In this way we can approximate the delay operator $\mathcal{A}$, defined by Definition 5.13, with the matrix $\mathbf{M} : X_d \to X_d$ with

$$\mathbf{Mz} = \begin{bmatrix} \mathcal{P}'(\theta_1) \\ \vdots \\ \mathcal{P}'(\theta_{d-1}) \\ A_0\mathcal{P}(0) + \sum_{i=1}^{m} A_i \mathcal{P}(-\tau_i) \end{bmatrix}.$$

The numerical methods for computing eigenvalues of the delay eigenvalue problem, presented in [18, 19, 20, 21, 22], use the Lagrange form

$$\mathcal{P}(\theta) = \sum_{i=0}^{d-1} z_{i+1} \ell_i(\theta),$$

in order to obtain an explicit expression for the matrix $\mathbf{M} \in \mathbb{R}^{dn \times dn}$. Next, the discretized linear eigenvalue problem

$$\mathbf{Mz} = \lambda \mathbf{z}, \tag{5.22}$$

is solved by the QR method.

By choosing the grid points in the spectral discretization of $\mathcal{A}$ as scaled and shifted Chebyshev extremal points it is proven in [19] that spectral accuracy is obtained, i.e., the approximation error is of order $O(d^{-d})$. Consequently, the individual eigenvalues of the matrix $\mathbf{M}$ converge very fast to the corresponding eigenvalues of delay operator $\mathcal{A}$.

A companion-type reformulation

A companion-type pencil with the same eigenvalues as the matrix $\mathbf{M}$ is proposed in [49]. Therefore, we choose the grid points as follows

$$\theta_i = \frac{\tau_m}{2}(\alpha_i - 1), \qquad i = 1, \ldots, d, \tag{5.23}$$

where the normalized grid points α_i, i.e., scaled and shifted to the interval $[-1, 1]$, are

$$\alpha_i = \cos\left(\frac{i\pi}{d}\right), \qquad i = 1, \ldots, d.$$

Next, we define the shifted and scaled Chebyshev polynomials as

$$\widehat{T}_i(\theta) := T_i\left(2\frac{\theta}{\tau_m} + 1\right), \qquad i = 0, 1, \ldots, \tag{5.24}$$

where T_i is the ith order Chebyshev polynomial of the first kind. The corresponding integration map L_d, as defined in Definition 5.5, is given by

$$L_d = \frac{\tau_m}{4}\begin{bmatrix} 2 & & & & & \\ 0 & \frac{1}{2} & & & & \\ -1 & 0 & \frac{1}{3} & & & \\ & -\frac{1}{2} & 0 & \frac{1}{4} & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\frac{1}{d-2} & 0 & \frac{1}{d} \end{bmatrix}. \tag{5.25}$$

By representing the interpolating polynomial $\mathcal{P}$ as follows

$$\mathcal{P}(\theta) = \sum_{i=0}^{d-1} c_{i+1}\widehat{T}_i(\theta_i),$$

where θ_i is given by (5.23) and $c_i \in \mathbb{C}^n$ for $i = 1, \ldots, d$, the discretized eigenvalue problem (5.22) is equivalent to

$$\widetilde{\mathbf{A}}\mathbf{x} = \lambda\widetilde{\mathbf{B}}\mathbf{x},$$

where

$$\widetilde{\mathbf{A}} = \begin{bmatrix} R_0 & R_1 & \cdots & R_{d-1} \\ & I & & \\ & & \ddots & \\ & & & I \end{bmatrix}, \qquad \widetilde{\mathbf{B}} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ & & & \\ & & \underline{L}_d^T & \\ & & & \end{bmatrix} \otimes I_n, \tag{5.26}$$

with $\underline{L}_d \in \mathbb{R}^{d\times(d-1)}$ the leading part of the integration matrix L_d,

$$R_j = A_0 + \sum_{i=0}^{m} A_i\widehat{T}_j(-\tau_i), \qquad j = 0, 1, \ldots, \tag{5.27}$$

and

$$\mathbf{x} := \begin{bmatrix} c_1 \\ \vdots \\ c_d \end{bmatrix}.$$

An infinite-dimensional operator setting equivalence

In order to show the equivalence of the spectral discretization approach with an infinite-dimensional operator setting, we use the results of §5.3.1 and represent the functions $\phi_1, \phi_2, \ldots$ in the shifted and scaled Chebyshev basis.

Let $p_i(\theta) = \widehat{T}_i(\theta)$ for $i = 0, 1, \ldots$ be given by (5.24). Then, $p_i(0) = \widehat{T}_i(0) = T_i(1) = 1$ for all $i \geq 0$ and (5.17)–(5.18) are respectively given by

$$\mathbf{A} := \left[\begin{array}{c|ccc} 1 & 1 & \cdots & 1 \\ \hline 0 & & I_{d-1} & \end{array}\right] \otimes I_n,$$

$$\mathbf{B} := \left[\begin{array}{ccccc} C_0 & C_1 & \cdots & C_{d-1} \\ \hline & & \underline{L}_d^T \otimes I_n & \end{array}\right],$$

where $\underline{L}_d \in \mathbb{R}^{d\times(d-1)}$ is the leading part of the integration matrix L_d (5.25) and

$$C_i = -A(0)^{-1}\left(A\left(\frac{d}{d\theta}\right)\int_0^\theta \widehat{T}_i(s)ds\right)(0), \qquad i = 0, 1, \ldots. \tag{5.28}$$

To prove the equivalence between the standard Arnoldi method applied to $\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{B}}$, where the spectral discretization matrices $\widetilde{\mathbf{A}}$ and $\widetilde{\mathbf{B}}$ are defined in (5.26), and the Delay Arnoldi method [49], i.e., the Infinite Arnoldi method applied to the inverse delay operator with a shifted and scaled Chebyshev scalar product, we now only need to show that for all $j < d$ holds

$$\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{B}}\mathbf{x} = \mathbf{A}^{-1}\mathbf{B}\mathbf{x},$$

where only the first j blocks of the vector $\mathbf{x} \in \mathbb{C}^{dn}$ are nonzero. Define

$$\mathbf{E} := \begin{bmatrix} R_0^{-1} & X_1 & \cdots & X_{d-1} \\ & I & & \\ & & \ddots & \\ & & & I \end{bmatrix},$$

where

$$X_j := I - R_0^{-1}R_j, \qquad j = 0, 1, \ldots,$$

with $R_0, R_1, \ldots$ defined by (5.27). Then, it immediately follows that $\mathbf{A} = \mathbf{E}\widetilde{\mathbf{A}}$. Next, note that left multiplication of $\widetilde{\mathbf{B}}$ by $\mathbf{E}$ only affects the first block row, which results in

$$\left(e_1^T \otimes I_n\right)\mathbf{E}\widetilde{\mathbf{B}} = R_0^{-1}\begin{bmatrix} I_n & \cdots & I_n \end{bmatrix} + \begin{bmatrix} X_1 & \cdots & X_{d-1} \end{bmatrix}\left(\underline{L}_d^T \otimes I_n\right).$$

We will now show that the first block row of $\mathbf{E}\widetilde{\mathbf{B}}$ is equal to the first block row of $\mathbf{B}$ except for the last block. Therefore, we rewrite (5.28) as follows

$$\begin{aligned}
&\begin{bmatrix} C_0 & \cdots & C_{d-1} \end{bmatrix} \\
&\quad = -A(0)^{-1} \left(A\left(\frac{d}{d\theta}\right) \begin{bmatrix} \int_0^\theta \widehat{T}_0(s)ds & \cdots & \int_0^\theta \widehat{T}_{d-1}(s)ds \end{bmatrix} \right)(0), \\
&\quad = -A(0)^{-1} \left(A\left(\frac{d}{d\theta}\right) \begin{bmatrix} \widehat{T}_1(\theta) - \widehat{T}_1(0) & \cdots & \widehat{T}_d(\theta) - \widehat{T}_d(0) \end{bmatrix} L_d^T \right)(0), \\
&\quad = -A(0)^{-1} \left(A\left(\frac{d}{d\theta}\right) \begin{bmatrix} \widehat{T}_1(\theta) - 1 & \cdots & \widehat{T}_d(\theta) - 1 \end{bmatrix} \right)(0) \cdot \left(L_d^T \otimes I_n\right), \\
&\quad = -A(0)^{-1} \left(A\left(\frac{d}{d\theta}\right) \begin{bmatrix} \widehat{T}_1(\theta) & \cdots & \widehat{T}_d(\theta) \end{bmatrix} \right)(0) \cdot \left(L_d^T \otimes I_n\right) \\
&\qquad\qquad + \begin{bmatrix} I_n & \cdots & I_n \end{bmatrix} \cdot \left(L_d^T \otimes I_n\right).
\end{aligned} \tag{5.29}$$

Since for any entire function f holds that

$$\begin{aligned}
A\left(\frac{d}{d\theta}\right) f(0) &= f'(0)I - A_0 f(0) - \sum_{i=1}^m A_i \left(f(0) - \frac{\tau_i}{1!} f'(0) + \frac{\tau_i^2}{2!} f''(0) - \cdots \right), \\
&= f'(0)I - A_0 f(0) - \sum_{i=1}^m A_i f(-\tau_i),
\end{aligned}$$

where $A(\lambda)$ is defined in (5.21), we have

$$A\left(\frac{d}{d\theta}\right) \widehat{T}_j(0) = \widehat{T}_j'(0)I - R_j, \qquad j = 0, 1, \ldots. \tag{5.30}$$

Now, substituting (5.30) in (5.29) and using that $R_0 = -A(0)$ yields

$$\begin{aligned}
\begin{bmatrix} C_0 & \cdots & C_{d-1} \end{bmatrix} &= R_0^{-1} \begin{bmatrix} \widehat{T}_1'(0)I_n - R_1 & \cdots & \widehat{T}_d'(0)I_n - R_d \end{bmatrix} \cdot \left(L_d^T \otimes I_n\right) \\
&\qquad + \begin{bmatrix} I_n & \cdots & I_n \end{bmatrix} \cdot \left(L_d^T \otimes I_n\right), \\
&= R_0^{-1} \begin{bmatrix} \widehat{T}_0(0)I_n & \cdots & \widehat{T}_{d-1}(0)I_n \end{bmatrix} \\
&\qquad + \begin{bmatrix} I_n - R_0^{-1}R_1 & \cdots & I_n - R_0^{-1}R_d \end{bmatrix} \cdot \left(L_d^T \otimes I_n\right), \\
&= R_0^{-1} \begin{bmatrix} I_n & \cdots & I_n \end{bmatrix} + \begin{bmatrix} X_1 & \cdots & X_d \end{bmatrix} \cdot \left(L_d^T \otimes I_n\right), \\
&= \left(e_1^T \otimes I_n\right) \mathbf{E}\widetilde{\mathbf{B}} + \begin{bmatrix} 0 & \cdots & 0 & \frac{1}{d}X_d \end{bmatrix}.
\end{aligned}$$

Thus, for any vector $\mathbf{x} \in \mathbb{C}^{dn}$ of which only the first $j < d$ blocks are nonzero, we have

$$\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{B}}\mathbf{x} = \widetilde{\mathbf{A}}^{-1}\mathbf{E}^{-1}\mathbf{E}\widetilde{\mathbf{B}}\mathbf{x} = \mathbf{A}^{-1}\mathbf{B}\mathbf{x}.$$

Therefore, we can conclude that the standard Arnoldi method applied to the spectral discretization in companion-type form [49] with a short starting vector is equivalent to the Infinite Arnoldi method applied to the inverse delay operator $\mathcal{A}^{-1}$ with the Chebyshev scalar product and a constant starting vector.

5.4 Numerical experiments

The Infinite Arnoldi method (Algorithm 5.2) with a scalar product in the monomial basis, i.e., the Taylor–Arnoldi method (Algorithm 3.1), has already been illustrated for a root-finding problem in §3.3.1. Recall also that the Taylor–Arnoldi method corresponds to the dynamic variant of the Fully Rational Krylov method with all interpolation nodes at zero and all poles at infinity. Here, we focus on solving delay eigenvalue problems by the Infinite Arnoldi method and illustrate that the choice of basis and scalar product can improve the convergence of the algorithm remarkably. Firstly, we consider a scalar delay problem in §5.4.1 for which we show the difference between a scalar product in the monomial basis, used in the Taylor–Arnoldi method, and a shifted and scaled Chebyshev scalar product, used in the Delay Arnoldi method. Next, we illustrate the Delay Arnoldi method for a large-scale delay problem in §5.4.2. Finally, we consider a delay eigenvalue problem with low rank in §5.4.3 for which we illustrate the advantage of applying the Infinite Arnoldi method to $\mathcal{F}\mathcal{A}^{-1}$ instead of to $\mathcal{A}^{-1}$.

All numerical experiments are performed in MATLAB version 7.14.0 (R2012a) on a Dell Latitude notebook running an Intel(R) Core(TM) i5-2540M CPU @ 2.60GHz quad core processor with 8 GB RAM. Our experiments can be reproduced with the publicly available code of [105].

5.4.1 Scalar delay problem

We start with a scalar delay eigenvalue problem [49]

$$A(\lambda) = \lambda - (2 - e^{-2}) - e^{-\lambda} = 0,$$

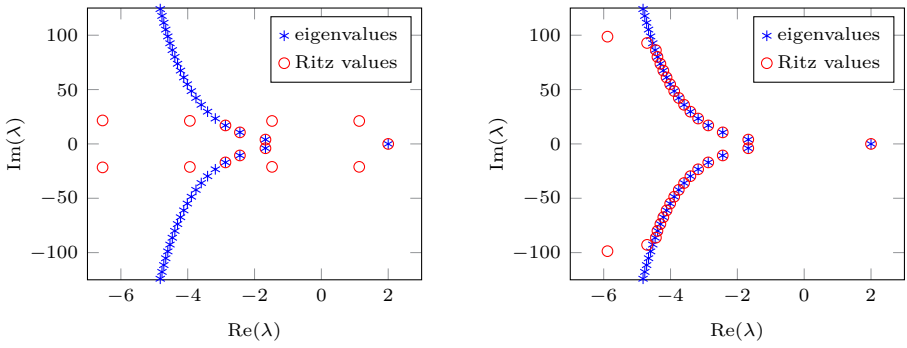
and suppose we are interested in the eigenvalues closest to the origin. For measuring the convergence of an approximate eigenvalue λ, we used the following relative residual norm

$$E(\lambda) = \frac{|A(\lambda)|}{|\lambda| + |2 - e^{-2}| + |e^{-\tau\lambda}|}.$$

We now compare the Taylor–Arnoldi method to the Delay Arnoldi method. Both methods can be seen as special cases of the Infinite Arnoldi method: the

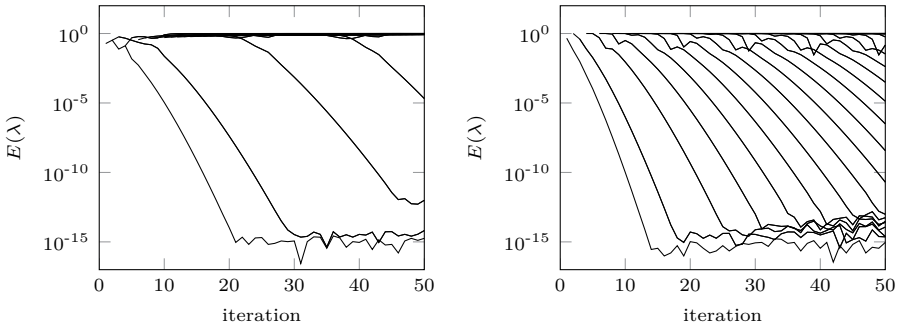
first one uses a scalar product in monomial basis, whereas the latter uses a shifted and scaled Chebyshev basis with a corresponding scalar product in this basis.

The computed Ritz values after 50 iterations of the Taylor–Arnoldi method and the Delay Arnoldi method are shown in Figure 5.1(a) and Figure 5.1(b), respectively. The corresponding convergence history is given in Figure 5.2. From these figures, we see that the choice of basis and corresponding scalar product in the Infinite Arnoldi method can have a significant impact on the convergence of Algorithm 5.2. For the Delay Arnoldi method, we even observe a



(a) Taylor–Arnoldi method (b) Delay Arnoldi method

Figure 5.1: Scalar delay eigenvalue problem: Ritz values after 50 iterations of (a) the Taylor–Arnoldi method and (b) the Delay Arnoldi method.



(a) Taylor–Arnoldi method (b) Delay Arnoldi method

Figure 5.2: Scalar delay eigenvalue problem: convergence history for (a) the Taylor–Arnoldi method and (b) the Delay Arnoldi method.

geometric convergence in Figure 5.2(b). This can be explained by its connection with the standard Arnoldi method applied to a pencil obtained by a spectral discretization of the delay operator. See [120] for the connection between a spectral discretization and rational approximation of exponential functions.

5.4.2 Large-scale delay problem

We consider a large-scale delay eigenvalue problem [49] corresponding to the delay differential equation

$$\frac{\partial v(x,t)}{\partial t} = \frac{\partial^2 v(x,t)}{\partial x^2} + a_0(x)v(x,t) + a_1(x)v(\pi - x, t-1), \tag{5.31}$$

where $a_0(x) = -2\sin(x)$, $a_1(x) = 2\sin(x)$, and $v_x(0,t) = v_x(\pi,t) = 0$. We discretize (5.31) by approximating the second derivative in space with central differences and obtain the following NLEP

$$A(\lambda)x = (\lambda I - A_0 - A_1 e^{-\tau\lambda})x = 0, \tag{5.32}$$

where $A_0, A_1 \in \mathbb{R}^{5000\times 5000}$ and $\tau = 1$. For measuring the convergence of an approximate eigenpair (λ, x), we used the following relative residual norm

$$E(\lambda, x) = \frac{\|A(\lambda)x\|_2 / \|x\|_2}{\|A_0\|_1 + |\lambda| + |e^{-\tau\lambda}|\|A_1\|_1}. \tag{5.33}$$

The delay eigenvalue problem (5.32) is again solved with the Delay Arnoldi method. The Ritz values after 100 iterations are shown in Figure 5.3(a). The corresponding convergence history is given in Figure 5.3(b). Note that after 100 iterations already more than 50 eigenvalues started to converge and 18 eigenvalues are converged up to machine precision.

5.4.3 Delay problem with low rank

We model a one-dimensional clamped beam and delayed feedback control localized at the endpoint with a partial delay differential equation. See [119, 41] for PDEs with delays. More precisely, we consider the following one-dimensional delay differential equation

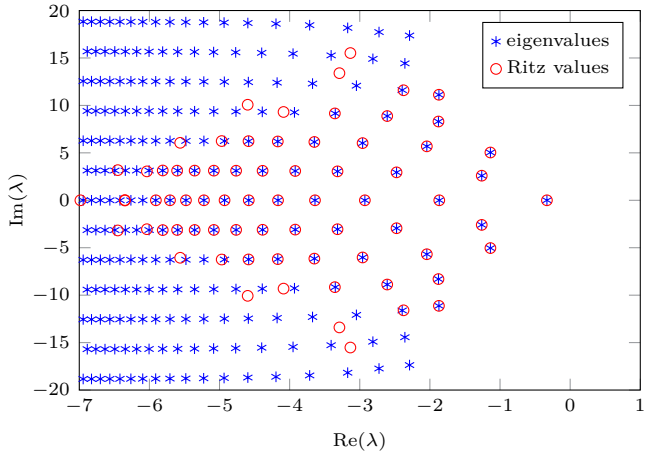
$$u_t(x,t) = u_{xx}(x,t) + \delta(x - 0.5)u(0.5, t-\tau),$$

with boundary conditions $u(0,t) = u_x(1,t) = 0$ and $\delta(x)$ a Dirac impulse such that the problem corresponds to delayed pointwise feedback at $x = 0.5$. The finite difference discretization results in the following delay eigenvalue problem [105]

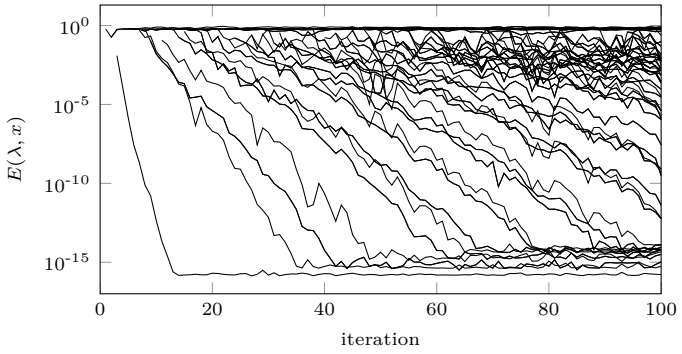
$$A(\lambda) = \lambda I + A_0 + A_1 e^{-\tau\lambda}, \tag{5.34}$$

where $A_0 \in \mathbb{C}^{n\times n}$ is a tridiagonal matrix and A_1 a rank 1 matrix. The goal in this experiment is to compute the 15 eigenvalues closest to the origin. For measuring the convergence of an approximate eigenpair, we used the relative residual norm (5.33).

The delay eigenvalue problem (5.34) with $n = 10.001$ is solved by the standard Delay Arnoldi method and by the Delay Arnoldi method with low rank exploitation. The eigenvalues are shown in Figure 5.4(a). An eigenvalue is classified as converged if $E(\lambda, x) \leq 10^{-10}$.



(a) eigenvalues

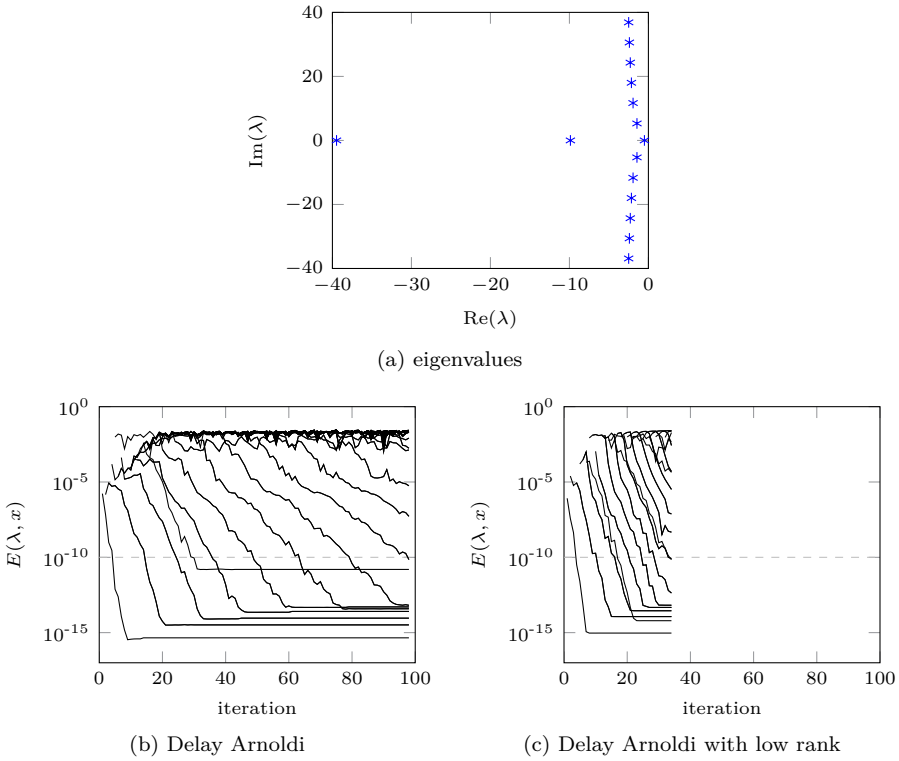


(b) convergence of residuals

Figure 5.3: Large-scale delay eigenvalue problem: (a) Ritz values after 100 iterations of the Delay Arnoldi method and (b) the corresponding convergence history.

The convergence history of the Delay Arnoldi method without and with low rank exploitation is given in Figure 5.4(b) and Figure 5.4(c), respectively. In these figures, we observe that the Delay Arnoldi method with low rank exploitation needs almost 3 times less iterations for computing the 15 eigenvalues closest to the origin. This illustrates once again that the choice of the scalar product, in this case the low rank shifted and scaled Chebyshev scalar product, can have a huge effect on the convergence of the eigenpairs.

Furthermore, the low rank exploitation reduces both the memory cost and the orthogonalization cost since the subspace vectors grow in every iteration of the standard Delay Arnoldi method with a block of size $n = 10.001$. On the other hand in the low rank version, they grow after the first iteration only with blocks of size $r = 1$. Therefore, the standard Delay Arnoldi method handles in this experiment with vectors of size $O(10^6)$, while the low rank version only deals with vectors of size $O(10^4)$.



(a) eigenvalues

(b) Delay Arnoldi

(c) Delay Arnoldi with low rank

Figure 5.4: Large-scale delay eigenvalue problem with low rank: (a) Eigenvalues, (b) convergence history for the Delay Arnoldi method, and (c) convergence history for the Delay Arnoldi method with low rank exploitation.

5.5 Conclusions

In this chapter, we presented an alternative approach to interpolation based techniques for solving nonlinear eigenvalue problems. Therefore, we first transform the nonlinear eigenvalue problem into an equivalent linear operator eigenvalue problem. Next, the eigenvalues of the operator $\mathcal{A}$ are computed.

The Infinite Arnoldi method is an Arnoldi method in a function setting applied to the operator $\mathcal{A}^{-1}$ for computing the smallest eigenvalues in magnitude of the nonlinear eigenvalue problem. By representing the functions in a degree-graded basis, by using a constant starting function, and by a particular scalar product, the structure of the infinite dimensional linear operator $\mathcal{A}^{-1}$ can be exploited such that the Infinite Arnoldi method only involves linear algebra operations applied to matrices of finite size.

We also proved that there is an equivalence between the Infinite Arnoldi method and the standard Arnoldi method applied to a large companion-type pencil. In case of the monomial basis together with a scalar product in this basis, the Infinite Arnoldi method is identical to the Taylor–Arnoldi method. For the delay eigenvalue problems, we illustrated that the Delay Arnoldi method, i.e., the Infinite Arnoldi method applied to the inverse delay operator with a shifted and scaled Chebyshev scalar product, is equivalent to the standard Arnoldi method applied to a pencil obtained by a spectral discretization of the delay operator.

The numerical examples have shown that the choice of basis together with its corresponding scalar product, can have a significant impact on the convergence of the Infinite Arnoldi method. Remark that the low rank exploitation also changes the operator and therefore the convergence properties as well. For delay eigenvalue problems, the operator setting in a shifted and scaled Chebyshev basis results in a very powerful algorithm, i.e., the Delay Arnoldi method. Its good convergence properties can to a large extent be attributed to the fact that the underlying approximation is a rational approximation [120]. Note that this rational approximation is implicit in the sense that it is induced by the spectral discretization of the delay operator, e.g., the poles cannot be freely chosen as in the Fully Rational Krylov method. However, for a general nonlinear eigenvalue problem, an *optimal* choice of the basis and corresponding scalar product is less clear.

Chapter 6

Compact Rational Krylov Method

In this chapter, we present the framework of Compact Rational Krylov (CORK) methods for solving the nonlinear eigenvalue problem (NLEP)

$$A(\lambda)x = 0, \tag{6.1}$$

where $\lambda \in \Omega \subseteq \mathbb{C}$, $A : \Omega \to \mathbb{C}^{n\times n}$ is analytic on Ω, and $n \gg 1$.

As illustrated in the previous chapters, linearization based methods are often used for solving (6.1). Note that all linearization pencils discussed in Chapters 2–5 can be written in a similar form, i.e., $\mathbf{L}(\lambda) = \mathbf{A} - \lambda\mathbf{B}$, where the parts below the first block rows of $\mathbf{A}$ and $\mathbf{B}$ have the following Kronecker structure: $M \otimes I_n$ and $N \otimes I_n$, respectively. However, the major disadvantage of linearization based methods for solving large-scale NLEPs is the growing memory and orthogonalization costs with the iteration count, i.e., in general they are proportional to the degree of the polynomial. Therefore, the CORK family of rational Krylov methods maximally exploits this structure such that the extra memory and orthogonalization costs due to the linearization of the original eigenvalue problem are negligible for large-scale problems.

This chapter is organized as follows. Firstly, we introduce in Section 6.1 a uniform framework for representing the linearization pencils presented in Chapters 2–5. Next, we propose the compact rational Krylov decomposition in Section 6.2. Then in Section 6.4, the compact rational Krylov algorithm is introduced. Section 6.5 illustrates the proposed CORK method with two large-scale numerical examples. Finally, the main conclusions are summarized in Section 6.6.

6.1 Linearization pencils

Most methods presented in the previous chapters for solving the NLEP consist of two steps. Firstly, the matrix-valued function $A(\lambda)$ in (6.1) is approximated by a function $P(\lambda)$, which is often a polynomial or a rational function. Secondly, linearization is used to transform the eigenvalue problem $P(\lambda)x = 0$ into a linear pencil with the same eigenvalues. We start with the following definitions.

Definition 6.1. *Let $P(\lambda)$ with $P : \mathbb{C} \to \mathbb{C}^{n\times n}$ be defined as follows*

$$P(\lambda) := \sum_{i=0}^{d-1} (A_i - \lambda B_i) f_i(\lambda), \tag{6.2}$$

where $A_i, B_i \in \mathbb{C}^{n\times n}$ and f_i are scalar functions of λ. We assume that $P(\lambda)$ is regular, i.e., $\det P(\lambda)$ does not vanish identically and that there exists a linear relation between the functions f_i

$$(M - \lambda N) f(\lambda) = 0, \tag{6.3}$$

with $M, N \in \mathbb{C}^{(d-1)\times d}$ and $f(\lambda) := \begin{bmatrix} f_0(\lambda) & f_1(\lambda) & \cdots & f_{d-1}(\lambda) \end{bmatrix}^T \neq 0$ for all λ. We also assume that the matrix $M - \lambda N$ is of rank $d-1$ for all λ.

Definition 6.2 (Structured pencil [107])**.** *Let $P(\lambda)$, A_i, B_i, f_i, M, and N be as in Definition 6.1. Then, we define the $dn \times dn$ linear pencil $\mathbf{L}(\lambda)$ as follows*

$$\mathbf{L}(\lambda) = \mathbf{A} - \lambda \mathbf{B}, \tag{6.4}$$

where

$$\mathbf{A} = \left[\begin{array}{cccc} A_0 & A_1 & \cdots & A_{d-1} \\ \hline & M \otimes I_n & & \end{array}\right], \qquad \mathbf{B} = \left[\begin{array}{cccc} B_0 & B_1 & \cdots & B_{d-1} \\ \hline & N \otimes I_n & & \end{array}\right]. \tag{6.5}$$

Note that Definition 6.2 covers many of the linearizations used in the literature. Table 6.1 gives an overview of polynomial bases (Chapters 2-3) such as the monomial basis [69], the orthogonal bases including the Chebyshev basis [3], the Lagrange basis [108], and the Newton basis [3]. Also linear rational bases (Chapter 4) are covered, e.g., the rational monomial basis [76], the rational Newton basis [44], etc. Moreover, the spectral discretization (Chapter 5) used in [49] also fits in the pencil (6.4)–(6.5). Also note that Fiedler linearizations do, in general, not satisfy this structure [34, 5].

The Kronecker structure of the linearization matrices $\mathbf{A}$ and $\mathbf{B}$, defined in (6.5), can be exploited for efficiently solving linear systems originating from the shift-and-invert step in Krylov methods. Therefore, we introduce the following block ULP factorization.

Table 6.1: Transformation of matrix polynomials of degree d into the form of Definition 6.1.

(a) Matrix polynomials of degree d

Basis	$P(\lambda)$	Basis functions
Monomial	$\sum_{i=0}^{d} P_i \lambda^i$	λ^i
Orthogonal	$\sum_{i=0}^{d} C_i p_i(\lambda)$	$\lambda p_i(\lambda) = \alpha_i p_{i+1}(\lambda) + \beta_i p_i(\lambda) + \gamma_i p_{i-1}(\lambda)$
Newton	$\sum_{i=0}^{d} D_i n_i(\lambda)$	$n_0(\lambda) := 1,\ n_i(\lambda) := \prod_{k=0}^{i-1}(\lambda - \sigma_k)$ for $i > 0$
Lagrange	$\sum_{i=0}^{d} F_i \ell_i(\lambda)$	$\ell_i(\lambda) := \ell(\lambda)\frac{w_i}{\lambda - \sigma_i}$

(b) A_i and B_i for matrix polynomials of degree d in the form of (6.2)

Basis	A_i	B_i
Monomial	$P_i \quad i = 0, 1, \ldots, d-1$	$\begin{cases} 0 & i < d-1 \\ -P_d & i = d-1 \end{cases}$
Orthogonal	$\begin{cases} C_i & i < d-2 \\ C_{d-2} - \frac{\gamma_{d-1}}{\alpha_{d-1}} C_d & i = d-2 \\ C_{d-1} - \frac{\beta_{d-1}}{\alpha_{d-1}} C_d & i = d-1 \end{cases}$	$\begin{cases} 0 & i < d-1 \\ -\frac{1}{\alpha_{d-1}} C_d & i = d-1 \end{cases}$
Newton	$\begin{cases} D_i & i < d-1 \\ D_{d-1} - \sigma_{d-1} D_d & i = d-1 \end{cases}$	$\begin{cases} 0 & i < d-1 \\ -D_d & i = d-1 \end{cases}$
Lagrange	$\begin{cases} \sigma_{i+1} F_i & i < d-1 \\ \sigma_d F_{d-1} + \sigma_{d-1} \frac{w_d}{w_{d-1}} F_d & i = d-1 \end{cases}$	$\begin{cases} F_i & i < d-1 \\ F_{d-1} + \frac{w_d}{w_{d-1}} F_d & i = d-1 \end{cases}$

(c) f_i and its linear relations for matrix polynomials of degree d in the form of (6.2)

Basis	$f_i(\lambda)$	Linear relations
Monomial	λ^i	$f_{i+1}(\lambda) = \lambda f_i(\lambda)$
Orthogonal	$p_i(\lambda)$	$\alpha_i f_{i+1}(\lambda) = (\lambda - \beta_i) f_i(\lambda) - \gamma_i f_{i-1}(\lambda)$
Newton	$n_i(\lambda)$	$f_{i+1}(\lambda) = (\lambda - \sigma_i) f_i(\lambda)$
Lagrange	$-\ell_i(\lambda)/(\lambda - \sigma_{i+1})$	$w_i(\lambda - \sigma_{i+2}) f_{i+1}(\lambda) = w_{i+1}(\lambda - \sigma_i) f_i(\lambda)$

Theorem 6.3 (Block ULP decomposition)**.** *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be defined by* (6.5)*. Then, for every* $\mu \in \mathbb{C}$ *there exists a permutation matrix* $\mathcal{P} \in \mathbb{C}^{d\times d}$ *such that the matrix* $(M_1 - \mu N_1) \in \mathbb{C}^{(d-1)\times(d-1)}$ *is invertible with*

$$M =: \begin{bmatrix} m_0 & M_1 \end{bmatrix} \mathcal{P}, \qquad\qquad N =: \begin{bmatrix} n_0 & N_1 \end{bmatrix} \mathcal{P}.$$

Moreover, the pencil $\mathbf{L}(\mu)$ *can be factorized as follows*

$$\mathbf{L}(\mu) = \mathbf{A} - \mu\mathbf{B} = \mathcal{U}\mathcal{L}(\mathcal{P} \otimes I_n),$$

where

$$\mathcal{L} = \begin{bmatrix} P(\mu) & 0 \\ (m_0 - \mu n_0) \otimes I_n & (M_1 - \mu N_1) \otimes I_n \end{bmatrix},$$
$$\mathcal{U} = \begin{bmatrix} \alpha^{-1} I_n & (\bar{\mathbf{A}}_1 - \mu\bar{\mathbf{B}}_1)\left((M_1 - \mu N_1)^{-1} \otimes I_n\right) \\ 0 & I_{(d-1)n} \end{bmatrix},$$

with the scalar $\alpha = e_1^T \mathcal{P} f(\mu) \neq 0$ *and*

$$\begin{bmatrix} A_0 & A_1 & \cdots & A_{d-1} \end{bmatrix} =: \begin{bmatrix} \bar{A}_0 & \bar{\mathbf{A}}_1 \end{bmatrix} (\mathcal{P} \otimes I_n),$$
$$\begin{bmatrix} B_0 & B_1 & \cdots & B_{d-1} \end{bmatrix} =: \begin{bmatrix} \bar{B}_0 & \bar{\mathbf{B}}_1 \end{bmatrix} (\mathcal{P} \otimes I_n).$$

Proof. Firstly, since $\operatorname{rank}(M - \mu N) = d - 1$ for all μ by Definition 6.1, we can always find a permutation matrix $\mathcal{P}$ such that $M_1 - \mu N_1$ is invertible. Next, except for the top left block, all blocks of $\mathbf{L}(\mu)(\mathcal{P}^T \otimes I_n) = \mathcal{U}\mathcal{L}$ follow immediately from Definition 6.2. Thus, we only need to prove that

$$\bar{A}_0 - \mu\bar{B}_0 = P(\mu)/\alpha + (\bar{\mathbf{A}}_1 - \mu\bar{\mathbf{B}}_1)\left((M_1 - \mu N_1)^{-1}(m_0 - \mu n_0) \otimes I_n\right). \quad (6.6)$$

From Definition 6.2, we have that $\exists g \in \mathbb{C}^d \setminus \{0\} : (M - \lambda N)g = 0$. By using $\mathcal{P}\mathcal{P}^T = \mathcal{P}^T\mathcal{P} = I_d$ and multiplying from the left with $(M_1 - \mu N_1)^{-1}$ yields

$$\begin{bmatrix} (M_1 - \mu N_1)^{-1}(m_0 - \mu n_0) & I_{d-1} \end{bmatrix} \mathcal{P} g = 0.$$

Next, solving this linear system results in

$$\mathcal{P} g = \begin{bmatrix} 1 \\ -(M_1 - \mu N_1)^{-1}(m_0 - \mu n_0) \end{bmatrix}, \quad (6.7)$$

where g is only defined up to a scalar. Hence, using (6.3) and (6.7) we obtain

$$\mathcal{P} f(\mu) = \alpha \begin{bmatrix} 1 \\ -(M_1 - \mu N_1)^{-1}(m_0 - \mu n_0) \end{bmatrix} \quad (6.8)$$

with $\alpha = e_1^T \mathcal{P} f(\mu)$. By definition we have the identity

$$P(\lambda) = (e_1^T \otimes I_n) \cdot (\mathbf{A} - \lambda\mathbf{B}) \cdot (f(\lambda) \otimes I_n). \quad (6.9)$$

Now, substituting (6.8) into (6.9) proves the equality in (6.6). □

The eigenpair connections between $P(\lambda)$ and $\mathbf{L}(\lambda)$ now follow directly from the block ULP decomposition of $\mathbf{L}(\lambda)$.

Corollary 6.4 (Structured eigenvectors)**.** *Let $P(\lambda)$ and $\mathbf{L}(\lambda)$ be defined by Definitions 6.1 and 6.2, respectively.*

1. *If $P(\lambda)$ is regular, then $\mathbf{L}(\lambda)$ is also regular.*
2. *If the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$, then the pair $(\lambda_\star, f(\lambda_\star) \otimes x)$ is an eigenpair of $\mathbf{L}(\lambda)$.*
3. *If the pair $(\lambda_\star, \mathbf{x})$ is an eigenpair of $\mathbf{L}(\lambda)$, then there exists a vector x such that $\mathbf{x} = f(\lambda_\star) \otimes x$ and the pair $(\lambda_\star, x)$ is an eigenpair of $P(\lambda)$.*

Proof. Part *1* of the proof immediately follows from the block ULP decomposition of $\mathbf{L}(\lambda)$. Next, we consider the following identity

$$(\mathbf{A} - \lambda\mathbf{B}) \cdot (f(\lambda) \otimes I_n) = e_1 \otimes P(\lambda). \tag{6.10}$$

Then, the proof of part *2* immediately follows from taking $\lambda = \lambda_\star$ and multiplying (6.10) from the right with $1 \otimes x$. For the proof of part *3* we start with

$$\mathbf{L}(\lambda_\star)\mathbf{x} = (\mathbf{A} - \lambda_\star\mathbf{B})\mathbf{x} = 0.$$

Next, from the second till the last block row we find that

$$((M - \lambda_\star N) \otimes I_n)\mathbf{x} = 0.$$

By choosing $x = (e_1^T\mathcal{P} \otimes I_n)\mathbf{x}/\alpha$, with $\alpha = e_1^T\mathcal{P}f(\lambda_\star) \neq 0$, and using (6.8) we obtain $\mathbf{x} = f(\lambda_\star) \otimes x$. Again evaluating (6.10) at $\lambda_\star$ and multiplying from the right with $1 \otimes x$ completes the proof. □

Remark 6.5. For almost all λ, the same permutation matrix $\mathcal{P}$ can be taken, since the proof only relies on the invertibility of $M_1 - \lambda N_1$. In all papers mentioned earlier [69, 3, 33, 108, 76, 44, 49], $\mathcal{P}$ is chosen equal to the identity matrix. However, in general, it may happen that by an unlucky choice of λ permutation is necessary.

The major difficulty of (rational) Krylov methods using linearizations for large n is the growing memory and orthogonalization costs with the iteration count. In general, they are proportional to the degree of the polynomial. This means, at iteration j, the storage cost of the iteration vectors is of order $d \cdot j$ vectors of size n and the orthogonalization cost is of order $d \cdot j^2$ scalar products of size n. However, we can exploit the Kronecker structure, mentioned above, such that the memory cost is only of order $d + j$ vectors of size n (Section 6.2) and the orthogonalization cost is of order $(d + j)j$ scalar products of size n (Section 6.3).

6.2 A compact rational Krylov decomposition

Consider the standard rational Krylov recurrence relation (1.13) with the matrices $\mathbf{A}$ and $\mathbf{B}$ as defined by (6.5). Then, as in the compact Arnoldi decomposition [99], we can represent the subspace in a compact form. Note that the idea of a compact representation of Arnoldi vectors was presented for the Chebyshev basis in [55, 123].

We subdivide $\mathbf{V}_{j+1} \in \mathbb{C}^{dn\times(j+1)}$ as follows

$$\mathbf{V}_{j+1} = \begin{bmatrix}\mathbf{V}_j & \mathbf{v}_{j+1}\end{bmatrix} = \begin{bmatrix} V_j^{[1]} & v_{j+1}^{[1]} \\ \vdots & \vdots \\ V_j^{[d]} & v_{j+1}^{[d]} \end{bmatrix},$$

where $V_j^{[i]} \in \mathbb{C}^{n\times j}$ and $v_{j+1}^{[i]} \in \mathbb{C}^n$ for $i = 1, \ldots, d$.

Definition 6.6. *The matrix $Q_j \in \mathbb{C}^{n\times r_j}$ is defined so that its columns form an orthonormal basis for the column space of the matrix $\begin{bmatrix} V_j^{[1]} & \cdots & V_j^{[d]} \end{bmatrix}$ with rank r_j.*

Using Definition 6.6, we can express $V_j^{[i]}$ as

$$V_j^{[i]} = Q_j U_j^{[i]}, \qquad i = 1, \ldots, d,$$

where $U_j^{[i]} \in \mathbb{C}^{r_j\times j}$. Thus, we have

$$\mathbf{V}_j = \begin{bmatrix} Q_j & & \\ & \ddots & \\ & & Q_j \end{bmatrix} \begin{bmatrix} U_j^{[1]} \\ \vdots \\ U_j^{[d]} \end{bmatrix} = (I_d \otimes Q_j)\mathbf{U}_j, \tag{6.11}$$

where

$$\mathbf{U}_j = \begin{bmatrix} U_j^{[1]} \\ \vdots \\ U_j^{[d]} \end{bmatrix} \in \mathbb{C}^{dr_j\times j}. \tag{6.12}$$

Since both $\mathbf{V}_j$ and $I_d \otimes Q_j$ are matrices with orthonormal columns, $\mathbf{U}_j$ also has orthonormal columns. Using the compact representation of $\mathbf{V}_j$ (6.11), the rational Krylov recurrence relation (1.13) yields the following compact rational Krylov recurrence relation

$$\mathbf{A}(I_d \otimes Q_{j+1})\mathbf{U}_{j+1}\underline{H}_j = \mathbf{B}(I_d \otimes Q_{j+1})\mathbf{U}_{j+1}\underline{K}_j. \tag{6.13}$$

In order to refer to the compact rational Krylov (CORK) decomposition (6.13), we introduce the CORK quadruple.

Definition 6.7 (CORK quadruple)**.** *The quadruple* $\left(Q_{j+1}, \mathbf{U}_{j+1}, \underline{H}_j, \underline{K}_j\right)$ *with* $Q_{j+1} \in \mathbb{C}^{n \times r_{j+1}}$, $\mathbf{U}_{j+1} \in \mathbb{C}^{dr_{j+1} \times (j+1)}$, *and* $\underline{H}_j, \underline{K}_j \in \mathbb{C}^{(j+1) \times j}$ *is called a CORK quadruple of order* j *for* $(\mathbf{A}, \mathbf{B})$*, defined by* (6.5)*, if*

1. *it satisfies the CORK recurrence relation* (6.13)*,*
2. Q_{j+1} *and* $\mathbf{U}_{j+1}$ *have orthonormal columns and* Q_{j+1} *is of full rank,*
3. $\underline{K}_j$ *and* $\underline{H}_j$ *are upper Hessenberg matrices with* $\underline{H}_j$ *unreduced, and*
4. *none of the* $\sigma_i = k_{i+1,i}/h_{i+1,i}$, $i = 1, \ldots, j$ *is an eigenvalue of* $(\mathbf{A}, \mathbf{B})$*.*

Lemma 6.8. *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be defined by* (6.5)*. Then, the solution of the linear system*

$$(\mathbf{A} - \sigma\mathbf{B})\mathbf{x} = \mathbf{B}\mathbf{y}, \tag{6.14}$$

can be computed as follows: one block of $\mathbf{x}$*, say* $x^{[p]}$*, is obtained from a system solve with* $P(\sigma)$*, while the other blocks of* $\mathbf{x}$ *are obtained as linear combinations of* $x^{[p]}$ *and the blocks of* $\mathbf{y}$*.*

Proof. The proof follows from the block ULP decomposition (Theorem 6.3) of $L(\sigma) = \mathbf{A} - \sigma\mathbf{B}$. The index p corresponds to the block column index of $\mathbf{A}$ and $\mathbf{B}$ that is permuted to the first block column of $\mathcal{U}(\sigma)\mathcal{L}(\sigma)$ by the permutation matrix $\mathcal{P} \otimes I_n$. From the UL factorization it can indeed be seen that the other blocks of $\mathbf{x}$ are linear combinations of blocks of $\mathbf{B}\mathbf{y}$ and $x^{[p]}$. This completes the proof. □

Lemma 6.8 now gives rise to Algorithm 6.1, where we use the block ULP decomposition in order to solve (6.14). This results in 1 matrix-vector operation followed by 2 block triangular system solves and a permutation.

The following theorems summarize how the matrices Q_j and $\mathbf{U}_j$ can easily be extended into Q_{j+1} and $\mathbf{U}_{j+1}$, respectively. The Kronecker structure of the pencil $(\mathbf{A}, \mathbf{B})$, defined in (6.5), also provides an upper bound for the rank r_j of Q_j.

Theorem 6.9. *Let* Q_j *be as in Definition 6.6. Then,*

$$\operatorname{span}\{Q_{j+1}\} = \operatorname{span}\left\{Q_j, v_{j+1}^{[p]}\right\},$$

where $v_{j+1}^{[p]}$ *is the pth block in Lemma 6.8.*

Algorithm 6.1: System solve: $\mathbf{x} = (\mathbf{A} - \sigma\mathbf{B})^{-1}\mathbf{B}\mathbf{y}$

1 Compute the right hand side of (6.14): $\mathbf{z} = \mathbf{B}\mathbf{y}$.
2 Solve the block upper triangular system:

$$z^{[1]} = z^{[1]} - (\bar{\mathbf{A}}_1 - \sigma\bar{\mathbf{B}}_1)\left((M_1 - \sigma N_1)^{-1} \otimes I\right) z^{[2,\ldots,d]}.$$

3 Solve block lower triangular system:

$$z^{[1]} = P(\sigma)^{-1} z^{[1]}, \tag{a}$$

$$z^{[2,\ldots,d]} = \left((M_1 - \sigma N_1)^{-1} \otimes I\right)\left(z^{[2,\ldots,d]} - ((m_0 - \sigma n_0) \otimes I)\, z^{[1]}\right). \tag{b}$$

4 Permute the blocks of $\mathbf{z}$:

$$\mathbf{x} = (\mathcal{P}^T \otimes I)\mathbf{z}.$$

Proof. By using Definition 6.6 and Lemma 6.8 with $\sigma = \sigma_j$, we have

$$\begin{aligned}
\operatorname{span}\{Q_{j+1}\} &= \operatorname{span}\left\{\begin{bmatrix} V_{j+1}^{[1]} & \cdots & V_{j+1}^{[d]} \end{bmatrix}\right\}, \\
&= \operatorname{span}\left\{Q_j, v_{j+1}^{[1]}, \ldots, v_{j+1}^{[d]}\right\}, \\
&= \operatorname{span}\left\{Q_j, v_{j+1}^{[p]}\right\},
\end{aligned}$$

which completes the proof. □

Theorem 6.10. *Let Q_j be defined by Definition 6.6. Then, we have*

$$r_j < d + j.$$

Proof. By Definition 6.6, we have $\operatorname{span}\{Q_1\} = \operatorname{span}\{v_1^{[1]}, \ldots, v_1^{[d]}\}$. Then, from Theorem 6.9 immediately follows that $r_j = \operatorname{rank}(Q_j) < d + j$, which concludes the proof. □

Theorem 6.11. *Let $\mathbf{U}_j \in \mathbb{C}^{dr_j \times j}$ be defined by* (6.12)*. Then, $\mathbf{U}_{j+1} \in \mathbb{C}^{dr_{j+1} \times (j+1)}$ takes the following form*

$$\mathbf{U}_{j+1} = \begin{bmatrix} (I_d \otimes I_{r_{j+1} \times r_j})\mathbf{U}_j & \mathbf{u}_{j+1} \end{bmatrix},$$

where $\mathbf{u}_{j+1} \in \mathbb{C}^{dr_{j+1}}$, or when $r_{j+1} > r_j$

$$U_{j+1}^{[i]} = \begin{bmatrix} U_j^{[i]} & u_{j+1}^{[i]} \\ 0_{1\times j} & \end{bmatrix}, \qquad i = 1, \ldots, d.$$

Proof. The proof follows immediately from the definition. □

6.3 Two-level orthogonalization

The compact representation of the rational Krylov subspace not only results in a CORK decomposition, but also yields the subdivision of the orthogonalization process into two levels.

We start with expressing the starting vector $\mathbf{v}_1$ in a compact form, i.e.,

$$\mathbf{v}_1 = (I_d \otimes Q_1)\mathbf{U}_1,$$

where $Q_1 \in \mathbb{C}^{n\times r_1}$ and $\mathbf{U}_1 \in \mathbb{C}^{dr_1}$, with $r_1 = \operatorname{rank}\left(\begin{bmatrix} v_1^{[1]} & \cdots & v_1^{[d]}\end{bmatrix}\right)$. This results in a CORK quadruple of order 0. Next, given a CORK quadruple of order $j-1$, we compute the CORK quadruple of order j. This results in a two-level orthogonalization process. Firstly, Q_j is expanded into Q_{j+1} (*first level*). Secondly, $\mathbf{U}_j$ is expanded into $\mathbf{U}_{j+1}$ and the Hessenberg matrices are updated to $\underline{H}_j$ and $\underline{K}_j$ (*second level*).

6.3.1 First level orthogonalization

From Theorem 6.9, we know that for expanding Q_j into Q_{j+1}, we need to compute $v_{j+1}^{[p]}$ and orthogonalize this vector against Q_j. Furthermore, the shift-and-invert step (Algorithm 1.3, step 4)

$$\widehat{\mathbf{v}} := (\mathbf{A}-\sigma_j\mathbf{B})^{-1}\mathbf{B}\mathbf{w}_j = (\mathbf{A}-\sigma_j\mathbf{B})^{-1}\mathbf{B}(I_d\otimes Q_j)\mathbf{U}_j t_j, \tag{6.15}$$

can easily be solved by Algorithm 6.1. However, since

$$\operatorname{span}\{Q_{j+1}\} = \operatorname{span}\left\{Q_j, v_{j+1}^{[p]}\right\} = \operatorname{span}\left\{Q_j, \widehat{v}^{[p]}\right\},$$

we only need to compute $\widehat{v}^{[p]}$. Therefore, step 3(b) in Algorithm 6.1 can be saved. Next, we orthogonalize $\widehat{v}^{[p]}$ against Q_j. Thus, let us denote

$$\widetilde{q} := \widehat{v}^{[p]} - Q_j Q_j^* \widehat{v}^{[p]}, \qquad \text{and} \qquad \delta = \|\widetilde{q}\|.$$

We first suppose that $\delta \neq 0$ and use $q_{j+1} = \widetilde{q}/\delta$ to expand Q_j into

$$Q_{j+1} = \begin{bmatrix} Q_j & q_{j+1}\end{bmatrix},$$

with $r_{j+1} = r_j + 1$. On the other hand, $\delta = 0$ implies that $\widehat{v}^{[p]}$ lies in the subspace spanned by Q_j. This situation is called *deflation* in SOAR [12]. In this case, we take $Q_{j+1} = Q_j$ and $r_{j+1} = r_j$.

6.3.2 Second level orthogonalization

Once Q_{j+1} is known, we still have to compute $\mathbf{u}_{j+1}$. We will show that $\mathbf{u}_{j+1}$, $\underline{H}_j$, and $\underline{K}_j$ can be computed from a rational Krylov step on small matrices. To see this, we need the following lemma.

Lemma 6.12. *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be defined by* (6.5) *and define* $\widetilde{\mathbf{A}}_{j+1}$ *and* $\widetilde{\mathbf{B}}_{j+1}$ *as follows*

$$\widetilde{\mathbf{A}}_{j+1} = (I_d \otimes Q_{j+1}^*) \begin{bmatrix} P(\sigma_j)^{-1} & 0 \\ 0 & I_{(d-1)n} \end{bmatrix} \mathbf{A}(I_d \otimes Q_{j+1}),$$
$$\widetilde{\mathbf{B}}_{j+1} = (I_d \otimes Q_{j+1}^*) \begin{bmatrix} P(\sigma_j)^{-1} & 0 \\ 0 & I_{(d-1)n} \end{bmatrix} \mathbf{B}(I_d \otimes Q_{j+1}).$$

Then, the shift-and-invert rational Krylov relation (1.12) *is equivalent with*

$$\left(\widetilde{\mathbf{A}}_{j+1} - \sigma_j \widetilde{\mathbf{B}}_{j+1}\right)^{-1} \widetilde{\mathbf{B}}_{j+1} \underline{\mathbf{U}}_j t_j = \mathbf{U}_{j+1} \underline{h}_j,$$

where $\underline{\mathbf{U}}_j$ *is such that* $(I_d \otimes Q_j)\mathbf{U}_j = (I_d \otimes Q_{j+1})\underline{\mathbf{U}}_j$.

Proof. Firstly, note that by using the block ULP decomposition of Theorem 6.3, we have that

$$\begin{aligned} &\widetilde{\mathbf{A}}_{j+1} - \sigma_j \widetilde{\mathbf{B}}_{j+1} \\ &\quad = \begin{bmatrix} \alpha I_{r_{j+1}} & Q_{j+1}^* P(\sigma_j)^{-1} (\bar{\mathbf{A}}_1 - \sigma_j \bar{\mathbf{B}}_1) \left((M_1 - \sigma_j N_1)^{-1} \otimes Q_{j+1}\right) \\ 0 & I_{(d-1)r_{j+1}} \end{bmatrix} \\ &\qquad \begin{bmatrix} I_{r_{j+1}} & 0 \\ (m_0 - \sigma_j n_0) \otimes I_{r_{j+1}} & (M_1 - \sigma_j N_1) \otimes I_{r_{j+1}} \end{bmatrix} (\mathcal{P} \otimes I_{r_{j+1}}), \end{aligned}$$

is always nonsingular. Next, rewriting relation (1.12) as follows

$$(\mathbf{A} - \sigma_j \mathbf{B})(I_d \otimes Q_{j+1})\mathbf{U}_{j+1} \underline{h}_j = \mathbf{B}(I_d \otimes Q_j)\mathbf{U}_j t_j,$$

and multiplying on the left by

$$(I_d \otimes Q_{j+1}^*) \begin{bmatrix} P(\sigma_j)^{-1} & 0 \\ 0 & I_{(d-1)n} \end{bmatrix},$$

yields

$$\left(\widetilde{\mathbf{A}}_{j+1} - \sigma_j \widetilde{\mathbf{B}}_{j+1}\right) \mathbf{U}_{j+1} \underline{h}_j = \widetilde{\mathbf{B}}_{j+1} \underline{\mathbf{U}}_j t_j.$$

Using that $\widetilde{\mathbf{A}}_{j+1} - \sigma_j \widetilde{\mathbf{B}}_{j+1}$ is invertible completes the proof. □

As a consequence of Lemma 6.12, $\mathbf{u}_{j+1}$ satisfies the following standard rational Krylov recurrence relation

$$\widetilde{\mathbf{A}}_{j+1}\mathbf{U}_{j+1}\underline{H}_j = \widetilde{\mathbf{B}}_{j+1}\mathbf{U}_{j+1}\underline{K}_j, \tag{6.16}$$

where the Hessenberg matrices $\underline{H}_j$ and $\underline{K}_j$ are the same as in the original CORK recurrence relation (6.13). Hence, the second level orthogonalization in each iteration of the CORK algorithm can be seen as a standard rational Krylov step (6.16) with the small matrices $\widetilde{\mathbf{A}}_{j+1}$ and $\widetilde{\mathbf{B}}_{j+1}$.

Note that, although $\widetilde{\mathbf{A}}_{j+1}$ and $\widetilde{\mathbf{B}}_{j+1}$ might change in every iteration, they always have the same Kronecker structure below the first block row as $\mathbf{A}$ and $\mathbf{B}$, respectively. Therefore, it will not be necessary to construct $\widetilde{\mathbf{A}}_{j+1}$ and $\widetilde{\mathbf{B}}_{j+1}$ explicitly, as we will explain in Section 6.4.

6.4 Algorithm

In this section, we introduce the family of compact rational Krylov methods applied to the linearization matrices $\mathbf{A}$ and $\mathbf{B}$ defined in (6.5). Next, we also discuss in §6.4.1 how the orthogonalization cost can significantly be reduced, in §6.4.2 how to restart the CORK method, and in §6.4.3 how to exploit low rank structure of the nonlinear part.

Based on Lemma 6.8, Theorems 6.9–6.11, and the two levels of orthogonalization, explained in Section 6.3, the compact rational Krylov method can efficiently be implemented. The corresponding CORK algorithm is outlined in Algorithm 6.2.

Before starting the rational Krylov iteration in Algorithm 6.2, we need to choose a starting vector (step 1). A first possibility is taking a randomly generated vector $v_0 \in \mathbb{C}^{nd}$. Then, using the economy-size QR decomposition of $\begin{bmatrix} v_0^{[1]} & \cdots & v_0^{[d]} \end{bmatrix} = \mathcal{Q}\mathcal{R}$ yields

$$Q_1 = \mathcal{Q} \in \mathbb{C}^{n\times d}, \qquad \mathbf{U}_1 = \operatorname{vec}(\mathcal{R}) \in \mathbb{C}^{d^2},$$

and $r_1 = d$. On the other hand, from Corollary 6.4 we know that the eigenvectors have a Kronecker structure. Therefore, we can also start Algorithm 6.2 with $v_0 = f \otimes q_0$, where $q_0 \in \mathbb{C}^n$ and $f \in \mathbb{C}^d$. Consequently, this results in

$$Q_1 = q_0/\|q_0\| \in \mathbb{C}^n, \qquad \mathbf{U}_1 = \|q_0\| f \in \mathbb{C}^d, \tag{6.17}$$

and $r_1 = 1$.

For methods with dynamically growing linearizations, such as the infinite Arnoldi method [52] and the Newton rational Krylov method [106], the structured starting vector (6.17) corresponds to taking $f := e_1$, with e_1 the first unit

Algorithm 6.2: Compact Rational Krylov method

1 Choose Q_1 and $\mathbf{U}_1$, where $Q_1^*Q_1 = I_{r_1}$ and $\mathbf{U}_1^*\mathbf{U}_1 = 1$.
for $j = 1, 2, \ldots$ **do**
2 Choose shift: σ_j.
First level orthogonalization:
3 Compute: $\widehat{v}^{[p]}$ via Algorithm 6.1.
4 Orthogonalize: $\widetilde{q} := \widehat{v}^{[p]} - Q_j Q_j^* \widehat{v}^{[p]}$.
5 Next vector: $q_{j+1} = \widetilde{q}/\|\widetilde{q}\|$.
Second level orthogonalization:
6 Update matrices: $U_j^{[i]} = \begin{bmatrix} U_j^{[i]} \\ 0 \end{bmatrix}$ for $i = 1, \ldots, d$.
7 Compute: $\widehat{\mathbf{u}}$ via Algorithm 6.1.
8 Orthogonalize: $\widetilde{\mathbf{u}} := \widehat{\mathbf{u}} - \mathbf{U}_j \mathbf{U}_j^* \widehat{\mathbf{u}}$.
9 Next vector: $\mathbf{u}_{j+1} = \widetilde{\mathbf{u}}/h_{j+1,j}$, where $h_{j+1,1} = \|\widetilde{\mathbf{u}}\|$.
10 Compute eigenpairs: (λ_i, s_i) and test for convergence.
end
11 Compute eigenvectors: $\mathbf{x}_i = (I_d \otimes Q_{j+1})\mathbf{U}_{j+1}\underline{H}_j s_i$.

vector. Also in cases where it is inappropriate to choose f as a unit vector, i.e., in Lagrange basis, the structured starting vector (6.17) is advantageous, since we only need to store one vector of dimension n instead of d vectors of dimension n.

In each iteration step j of Algorithm 6.2 we start with choosing a shift σ_j (step 2). Next, the two levels of orthogonalization are performed in steps 3–5 and steps 6–9, respectively. In the first level, we compute the pth block $\widehat{v}^{[p]}$ of the next rational Krylov vector $\widehat{\mathbf{v}}$ by Algorithm 6.1. Note that, for only computing $\widehat{v}^{[p]}$, we can skip step 3(b). Next, we orthogonalize this vector $\widehat{v}^{[p]}$ against Q_j in order to obtain q_{j+1}. Thereafter, in the second level, we perform a standard rational Krylov step with the projected matrices $\widetilde{\mathbf{A}}_{j+1}$ and $\widetilde{\mathbf{B}}_{j+1}$ in order to obtain $\mathbf{u}_{j+1}$ and to expand the Hessenberg matrices. However, in practice, it is not necessary to form the matrices $\widetilde{\mathbf{A}}_{j+1}$ and $\widetilde{\mathbf{B}}_{j+1}$, since the pth block of $\widehat{\mathbf{u}}$ can be computed as follows

$$\widehat{u}^{[p]} = Q_{j+1}^* \widehat{v}^{[p]},$$

where $Q_j^* \widehat{v}^{[p]}$ is already computed during the first level orthogonalization. Then, the other blocks of $\widehat{\mathbf{u}}$ can be computed by Algorithm 6.1 where we skip steps 2 and 3(a). Finally, in step 10 of Algorithm 6.2, we compute the Ritz pairs (λ_i, s_i) and test for convergence.

Remark also that, from the definition of $\mathbf{U}$ (6.12), it is natural to represent $\mathbf{U}$ as a tensor. Therefore, in the implementation of Algorithm 6.2, we have stacked the blocks $U^{[i]}$ for $i = 1, \ldots, d$ behind each other.

6.4.1 Orthogonalization cost

The CORK method not only results in a much lower memory cost, but also the orthogonalization cost can be significantly reduced. In particular, we do not need to explicitly compute the other blocks than $\widehat{v}^{[p]}$ in (6.15), since the orthogonalization process in the second level only uses small matrices.

As mentioned before, the second level orthogonalization involves only 1 extra scalar product between vectors of size n for computing $\widehat{u}^{[p]}$. The other blocks of $\widehat{\mathbf{u}}$ can be computed as linear combinations of the blocks of $\mathbf{u}_j$ and $\widehat{u}^{[p]}$. This means, we only have to deal with vectors of length r_{j+1} in the second level orthogonalization. Therefore, the dominant orthogonalization cost occurs in the first level orthogonalization where the vector $\widehat{v}^{[p]} \in \mathbb{C}^n$ is orthogonalized against $q_1, \ldots, q_{r_j}$. In the second level orthogonalization we only have to deal with short vectors.

Table 6.2 gives an overview of the number of scalar products between vectors of size n in the orthogonalization process of the standard rational Krylov method and the CORK method. For dynamically growing linearization, such as in the Infinite Arnoldi method [49] and the Newton rational Krylov method [106], this number is of order $O(j^3)$. However, by using the CORK method with $v_1 = 1 \otimes x$ as starting vector, it reduces to $O(j^2)$. On the other hand, in the standard rational Krylov method for fixed size linearizations this number is of order $O(dj^2)$. Using the CORK method with a full starting vector v_1 it reduces to $O(j(d+j))$ and by using a structured starting vector $v_1 = f \otimes q_0$ it further reduces to $O(j^2)$. Note that this is the same order of magnitude as for the CORK method for dynamically growing linearizations.

Table 6.2: Number of scalar products between vectors of size n in the standard rational Krylov method and the CORK methods.

Linearization	Rat. Krylov	CORK	
		full v_1	$v_1 = f \otimes q_0$
dynamically growing	$O(j^3)$	-	$O(j^2)$
fixed size	$O(dj^2)$	$O(j(d+j))$	$O(j^2)$

6.4.2 Implicit restarting

Since the CORK method is a special variant of the rational Krylov method, we can also perform implicit restarting [63, 75] on Algorithm 6.2. Therefore, we first apply a transformation on the Hessenberg matrices $\underline{H}$ and $\underline{K}$, which allows to reorder and lock Ritz values. Next, representing the new subspace in its compact form, completes the restart of the compact rational Krylov process. Note that the restarting techniques explained in this section are a kind of generalization of the ones in [55] to rational Krylov and also to structured linearization pencils in different bases.

Suppose that after m iterations of the CORK algorithm, we have the CORK quadruple $(Q_{m+1}, \mathbf{U}_{m+1}, \underline{H}_m, \underline{K}_m)$ which we want to reduce to a smaller CORK quadruple $(Q_{k+1}, \mathbf{U}_{k+1}, \underline{H}_k, \underline{K}_k)$ with $k < m$. Therefore, we start with defining the following matrices

$$\underline{H}^+ = Y^* \underline{H}_m Z,$$
$$\underline{K}^+ = Y^* \underline{K}_m Z,$$

where $Y \in \mathbb{C}^{(m+1)\times(k+1)}$ and $Z \in \mathbb{C}^{m\times k}$ have unitary columns. With a proper choice of Y and Z (see [29, 85]), we have that

$$\mathbf{A}(I_d \otimes Q_{m+1})\mathbf{W}\underline{H}^+ = \mathbf{B}(I_d \otimes Q_{m+1})\mathbf{W}\underline{K}^+, \tag{6.18}$$

where $\mathbf{W} := \mathbf{U}_{m+1}Y$.

Next, note that in (6.18) the matrix Q remains the same, although the unwanted Ritz values are removed from the pencil (K, H). However, from Definition 6.7 and Theorem 6.10 we know that the rank of Q is bounded, also after restarting. Therefore, suppose the economy size singular value decomposition of

$$\begin{bmatrix} W^{[1]} & \cdots & W^{[d]} \end{bmatrix} = \mathcal{U}\mathcal{S}\begin{bmatrix} \mathcal{V}^{[1]} & \cdots & \mathcal{V}^{[d]} \end{bmatrix}, \tag{6.19}$$

where $\mathcal{U} \in \mathbb{C}^{r_{m+1}\times r}$, $\mathcal{S} \in \mathbb{C}^{r\times r}$, and $\mathcal{V}^{[i]} \in \mathbb{C}^{r\times(k+1)}$ for $i = 1, \ldots, d$. By definition of $\mathbf{W}$, we have $r \leq d + k$. Then, by substituting (6.19) into (6.18), we obtain

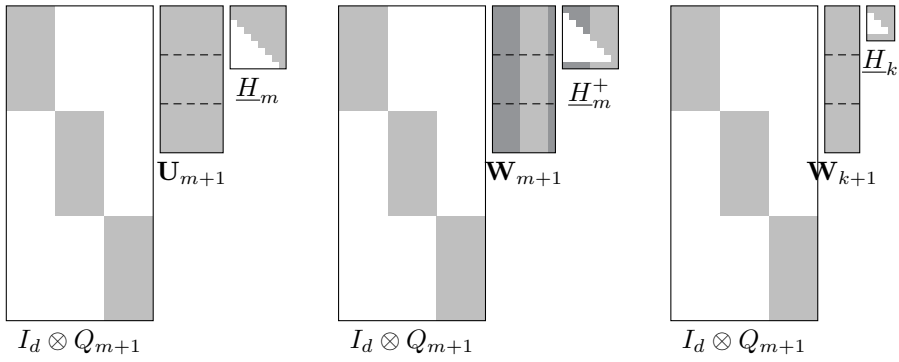
$$\mathbf{A}(I_d \otimes Q^+)\mathbf{U}^+\underline{H}^+ = \mathbf{B}(I_d \otimes Q^+)\mathbf{U}^+\underline{K}^+, \tag{6.20}$$

where

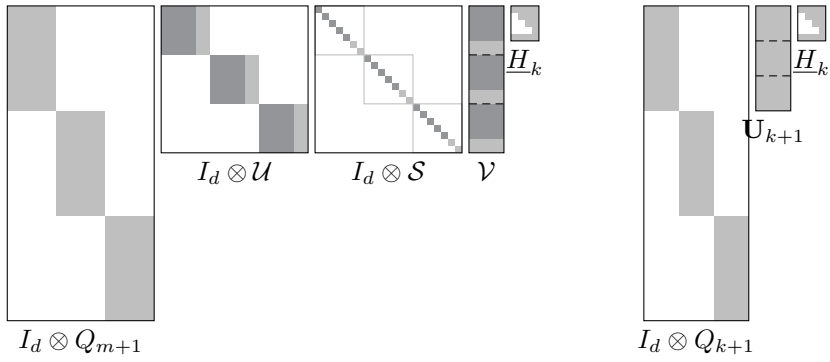
$$Q^+ := Q_{m+1}\mathcal{U}, \qquad \mathbf{U}^+ := \begin{bmatrix} \mathcal{S}\mathcal{V}^{[1]} \\ \vdots \\ \mathcal{S}\mathcal{V}^{[d]} \end{bmatrix}.$$

Finally, note that the recurrence relation (6.20) is a standard CORK recurrence relation of order k with $Q_{k+1} := Q^+$, $\mathbf{U}_{k+1} := \mathbf{U}^+$, $\underline{H}_k := \underline{H}^+$, and $\underline{K}_k := \underline{K}^+$.

The complete implicit restarting process is graphically illustrated in Figure 6.1. In this figure, we only illustrate the process for obtaining Q_{k+1}, $\mathbf{U}_{k+1}$, and $\underline{H}_k$ from Q_{m+1}, $\mathbf{U}_{m+1}$, and $\underline{H}_m$. However, the process is identical for $\underline{K}_m$. Figure 6.1(a) shows the deflation phase which consists of 2 steps. We start from the CORK quadruple of order m, shown in the left, and transform the Hessenberg matrices $\underline{H}$ and $\underline{K}$ into ordered partial Schur form, see in the middle. Next, we deflate the unwanted Ritz values and obtain $\mathbf{W}_{k+1}$ and $\underline{H}_k$. Remark that until now the matrix Q_{m+1} remains unchanged. Figure 6.1(b) shows the purging phase which consists of only 1 step. By computing an "economy size" singular value decomposition of the horizontally stacked blocks $\mathbf{W}^{[1,\ldots,d]}$, we obtain the reduced Q_{k+1}. After all these steps, we obtain the reduced CORK quadruple of order k which we will expand again in next rational Krylov steps.



(a) deflation



(b) purging

Figure 6.1: Graphical illustration of implicit restarting the CORK algorithm: (a) deflation and (b) purging.

6.4.3 Low rank exploitation

In several applications many of the blocks A_i and B_i in (6.5) are of low rank. Therefore, we generalize in this section the low rank structure exploitation proposed in [106].

Suppose that for $\widetilde{d} \leq i < d$ the blocks A_i and B_i in (6.5) are of low rank. Furthermore, we assume that

$$A_i = \widetilde{A}_i \widetilde{Z}^*, \qquad B_i = \widetilde{B}_i \widetilde{Z}^*, \qquad\qquad i = \widetilde{d}, \ldots, d-1,$$

where $\widetilde{A}_i, \widetilde{B}_i, \widetilde{Z} \in \mathbb{C}^{n \times \widetilde{n}}$, $\widetilde{Z}$ has orthonormal columns, and $\widetilde{n} \ll n$. Then, we can transform (6.5) into a linear companion pencil of dimension $\widetilde{d}n + (d-\widetilde{d})\widetilde{n}$

$$\widetilde{L}(\lambda) = \widetilde{\mathbf{A}} - \lambda \widetilde{\mathbf{B}}, \tag{6.21}$$

where

$$\widetilde{\mathbf{A}} = \left[\begin{array}{ccc|ccc} A_0 & \cdots & A_{\widetilde{d}-1} & \widetilde{A}_{\widetilde{d}} & \cdots & \widetilde{A}_{d-1} \\ \hline \multicolumn{3}{c|}{M_{11} \otimes I_n} & \multicolumn{3}{c}{0} \\ \multicolumn{3}{c|}{M_{21} \otimes \widetilde{Z}^*} & \multicolumn{3}{c}{M_{22} \otimes I_{\widetilde{n}}} \end{array}\right], \tag{6.22}$$

$$\widetilde{\mathbf{B}} = \left[\begin{array}{ccc|ccc} B_0 & \cdots & B_{\widetilde{d}-1} & \widetilde{B}_{\widetilde{d}} & \cdots & \widetilde{B}_{d-1} \\ \hline \multicolumn{3}{c|}{N_{11} \otimes I_n} & \multicolumn{3}{c}{0} \\ \multicolumn{3}{c|}{N_{21} \otimes \widetilde{Z}^*} & \multicolumn{3}{c}{N_{22} \otimes I_{\widetilde{n}}} \end{array}\right], \tag{6.23}$$

with the assumption that $M_{12} = N_{12} = 0$. This is the case for many of the bases used in the literature, e.g., degree-graded polynomial bases, Lagrange basis, rational Newton basis, etc. Similar to Corollary 6.4, the linearization (6.21)–(6.23) yields the following structured eigenvector

$$\widetilde{\mathbf{x}} = \left[\begin{array}{c} f_1 \otimes x \\ \hline f_2 \otimes \widetilde{Z}^* x \end{array}\right],$$

which is the motivation to subdivide $\mathbf{V}$ as follows

$$\mathbf{V} = \left[\begin{array}{c} V^{[1]} \\ \vdots \\ V^{[\widetilde{d}]} \\ \hline \widetilde{V}^{[\widetilde{d}+1]} \\ \vdots \\ \widetilde{V}^{[d]} \end{array}\right] = \left[\begin{array}{ccc|ccc} Q & & & & & \\ & \ddots & & & & \\ & & Q & & & \\ \hline & & & \widetilde{Q} & & \\ & & & & \ddots & \\ & & & & & \widetilde{Q} \end{array}\right] \left[\begin{array}{c} U^{[1]} \\ \vdots \\ U^{[\widetilde{d}]} \\ \hline \widetilde{U}^{[\widetilde{d}+1]} \\ \vdots \\ \widetilde{U}^{[d]} \end{array}\right],$$

where

$$V^{[1,\ldots,\widetilde{d}]} \in \mathbb{C}^{n\times j}, \qquad Q \in \mathbb{C}^{n\times r}, \qquad \mathbf{U} \in \mathbb{C}^{r\times j},$$
$$V^{[\widetilde{d}+1,\ldots,d]} \in \mathbb{C}^{\widetilde{n}\times j}, \qquad \widetilde{Q} \in \mathbb{C}^{\widetilde{n}\times\widetilde{r}}, \qquad \widetilde{\mathbf{U}} \in \mathbb{C}^{\widetilde{r}\times j}.$$

In applications where $\widetilde{d}$ is relatively small compared to d, the memory cost can significantly be reduced since $r \leq \widetilde{d}+j$. Note that, due to the appearance of $\widetilde{Z}^*$ in (6.22) and (6.23), $\widetilde{r}$ is not bounded any more by Theorem 6.10. However, for large n the memory cost for storing $\widetilde{Q}$ (involving $\widetilde{n}$ and $\widetilde{r}$) is almost negligible compared to the one for Q (involving n and r). Furthermore, as we will illustrate in the numerical experiments in Section 6.5.2, $\widetilde{r}$ also remains bounded in practice.

6.5 Numerical experiments

We now illustrate the CORK method (Algorithm 6.2) with two large-scale examples. In the first example, we consider the delay eigenvalue problem for which we used a dynamically growing linearization based on spectral discretization. Consequently, the CORK method is equal to the Infinite Arnoldi method in which the subspace is represented in a compact form. In this example, we compare the memory usage and the orthogonalization cost for the Infinite Arnoldi method to the CORK method. We show results for both without and with the implicit restarting technique explained in §6.4.2. In the second example, we consider the `'gun'` problem for which we used a rational Newton linearization of fixed degree. Hence, the CORK method is equal to the static variant of the Fully Rational Krylov method in which the subspace is represented in a compact form. Here, we also use the CORK method with the low rank exploitation of §6.4.3.

All numerical experiments are performed in MATLAB version 7.14.0 (R2012a) on a Dell Latitude notebook running an Intel(R) Core(TM) i5-2540M CPU @ 2.60GHz quad core processor with 8 GB RAM. Our experiments can be reproduced with the publicly available code of [107].

6.5.1 Delay problem

We reconsider the large-scale delay eigenvalue problem of §5.4.2 corresponding to the delay differential equation

$$\frac{\partial v(x,t)}{\partial t} = \frac{\partial^2 v(x,t)}{\partial x^2} + a_0(x)v(x,t) + a_1(x)v(\pi - x, t-1), \tag{6.24}$$

where $a_0(x) = -2\sin(x)$, $a_1(x) = 2\sin(x)$, and $v_x(0,t) = v_x(\pi,t) = 0$. We discretize (6.24) by approximating the second derivative in space with central differences and obtain the following NLEP

$$A(\lambda)x = (A_0 - \lambda I + A_1 e^{-\tau\lambda})x = 0, \tag{6.25}$$

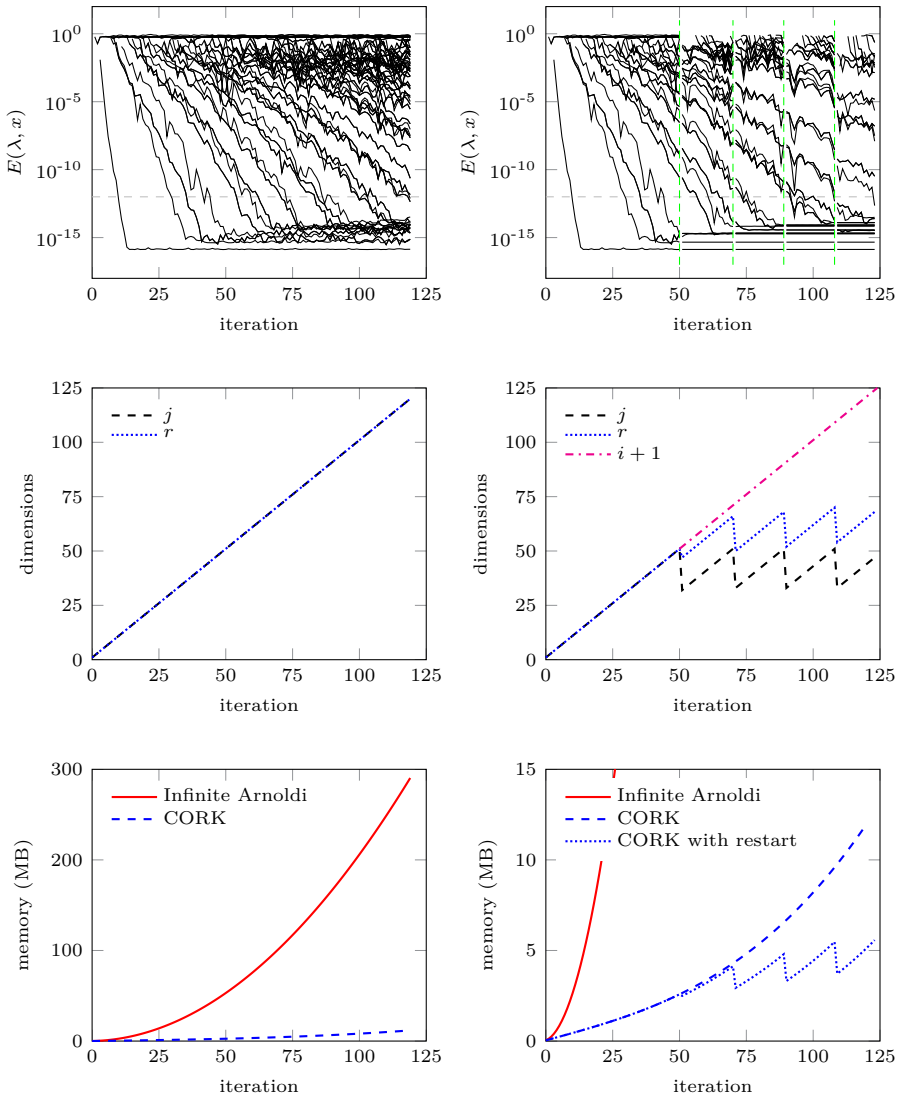
where $A_0, A_1 \in \mathbb{R}^{5000\times 5000}$ and $\tau = 1$. We again used the spectral discretization of §5.3.3 such that the linearization matrices have a companion-type structure.

The goal in this experiment is to compute the 20 eigenvalues closest to the origin. For measuring the convergence of an approximate eigenpair (λ, x), we used the following relative residual norm

$$E(\lambda, x) = \frac{\|A(\lambda)x\|_2 / \|x\|_2}{\|A_0\|_1 + |\lambda| + |e^{-\tau\lambda}|\|A_1\|_1}.$$

We first solved the NLEP (6.25) by Algorithm 6.2 without restart and chose a *short* starting vector $v_0 \in \mathbb{R}^{5000}$ such that $r_1 = \operatorname{rank}(Q_1) = 1$. Note that in this case the CORK method corresponds to the Infinite Arnoldi method where the subspace is represented in its compact form. The results of this experiment are shown in Figure 6.2(a). The convergence histories of the eigenpairs are given in the top figure, from which we see that after 119 iterations we found the 20 smallest eigenvalues in magnitude up to a tolerance of 10^{-12}. Since we did not use restart, we see in the middle figure that the rank of Q, r, and the dimension of the the subspace, j, increases with the iteration count i. The bottom figure shows the memory usage for storing the subspace in the Infinite Arnoldi method (Algorithm 5.2) and in the CORK method (Algorithm 6.2). From this figure, we see that a gain factor of 25 is achieved since the standard rational Krylov method requires $O(n \cdot i^2/2)$ to store the subspace and the CORK method only $O(n \cdot r)$, where $r = i + 1$.

Next, we solved (6.25) by Algorithm 6.2 combined with the implicit restarting technique explained in Section 6.4.2. Here, we chose the maximum dimension of the subspace, $m = 50$, and the number of selected Ritz values, $p = 30$. The results are shown in Figure 6.2(b). This figure again shows the convergence histories of the eigenpairs at the top. We see that the method requires 4 restarts (indicated by the vertical green dashed lines) and 123 iterations before finding the 20 smallest eigenvalues in magnitude up to the tolerance. In theory the rank of Q is unbounded, since we used a dynamically growing linearization and thus $r \leq i + 1$ with i the iteration count, i.e., the number of rational Krylov steps which is larger than m in the situation with restarting. However, we notice in the middle figure that in practice r stagnates in the course of the algorithm. Consequently, it has also a positive effect on the memory requirements as illustrated in the bottom figure. By using a restarted CORK method, we were able to reduce the memory cost for this delay problem by a factor 50.



(a) CORK method without restart

(b) CORK method with restart

Figure 6.2: Results for the delay problem.

6.5.2 Gun problem

We reconsider the 'gun' problem, see §1.1.2, which is a large-scale problem that models a radio-frequency gun cavity and is of the form [68]

$$A(\lambda)x = \Big(K - \lambda M + i\sqrt{\lambda - \sigma_1^2}\, W_1 + i\sqrt{\lambda - \sigma_2^2}\, W_2\Big)x = 0, \tag{6.26}$$

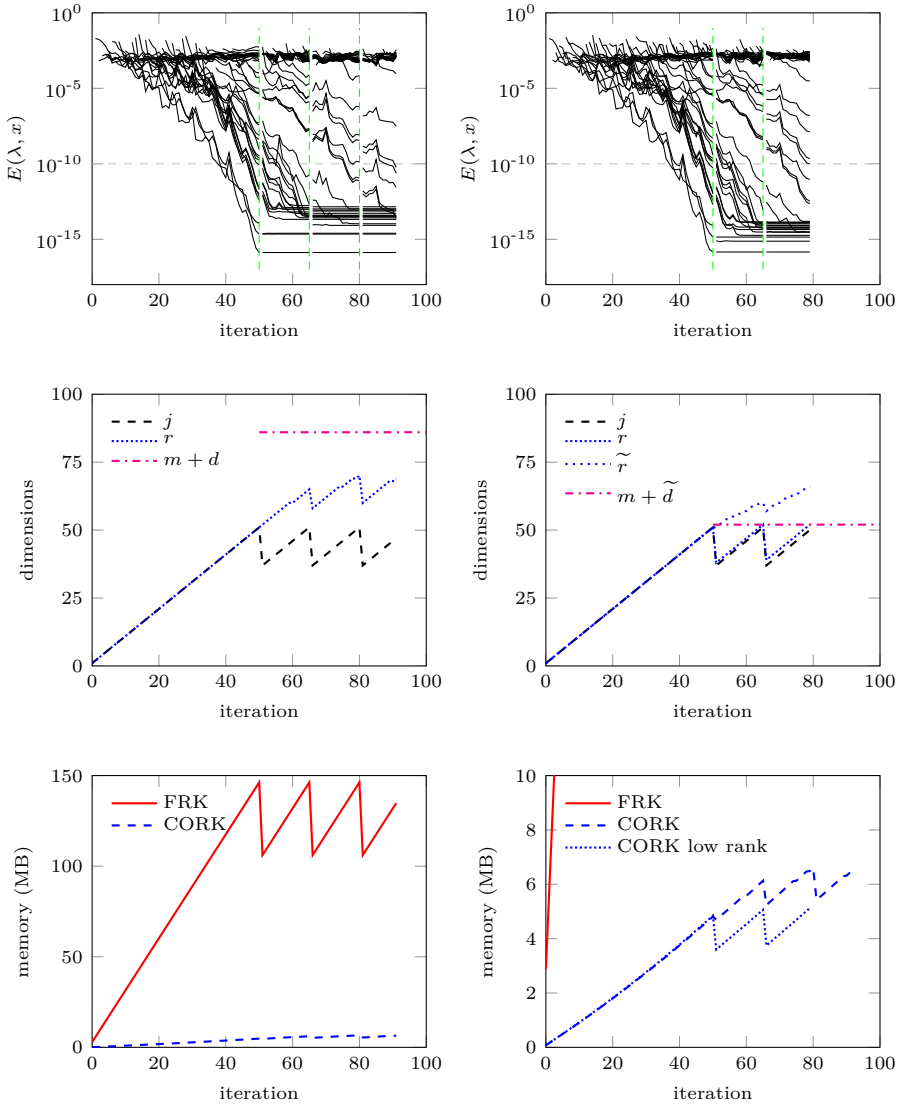
where M, K, W_1 and W_2 are real symmetric matrices of size 9956×9956, K is positive semidefinite, M is positive definite, and $\operatorname{rank}(W_1) + \operatorname{rank}(W_2) = 84$. The complex square root $\sqrt{\cdot}$ corresponds to the principal branch and as in [14], we take $\sigma_1 = 0$ and $\sigma_2 = 108.8774$. We used the same interpolating rational Newton polynomial of degree $d = 36$ as in the static variant of the Fully Rational Krylov method to approximate $A(\lambda)$ in (6.26).

The goal in this experiment is to compute the 20 eigenvalues closest to 250^2. For measuring the convergence of an approximate eigenpair (λ, x), we used the relative residual norm [68]

$$E(\lambda, x) = \frac{\|A(\lambda)x\|_2/\|x\|_2}{\|K\|_1 + |\lambda|\|M\|_1 + \sqrt{|\lambda - \sigma_1^2|}\|W_1\|_1 + \sqrt{|\lambda - \sigma_2^2|}\|W_2\|_1}.$$

In a first experiment, the NLEP (6.26) is solved by Algorithm 6.2 with maximum subspace dimension $m = 50$ and $p = 35$ selected Ritz values in every restart. We also used cyclically repeated shifts in the rational Krylov steps, indicated by "$\circ$" in Figure 4.6(a). The results of this experiment are shown in Figure 6.3(a). From the convergence histories of the eigenpairs in the top figure, we see that we need 91 iterations to compute the 20 eigenvalues closest to 250^2 up to a tolerance of 10^{-10}. In the middle figure, we see that the rank of Q, r, stagnates again in practice and remains significantly below the theoretical upper bound $r \leq m + d$. Next, comparing the subspace memory cost of the CORK method to the one of the Fully Rational Krylov method results in a reduction with a factor of more than 20.

In a final experiment, we solve (6.26) by Algorithm 6.2 and use the low rank exploitation of §6.4.3. Note that in this case, the blocks A_i and B_i for $i \geq \widetilde{d} = 2$ in the linearization pencil have only $\widetilde{n} = 84$ columns. For this experiment, we chose all parameters equal to the ones in the previous experiment. The corresponding results are shown in Figure 6.3(b). In the top figure with the convergence histories of the eigenpairs, we see that now only 79 iterations are needed to compute the 20 eigenvalues closest to 250^2. In the middle figure, we see that the rank of Q, r, is now bounded by the upper bound $r \leq m + \widetilde{d}$ with $\widetilde{d} < d$. Note also that the rank of $\widetilde{Q}$, $\widetilde{r}$, is small. Consequently, since in the CORK method with low rank exploitation the memory cost is dominated by Q with $r \leq m + \widetilde{d}$ compared to $r \leq m + d$ for the standard CORK method, we notice in the bottom figure that the subspace memory cost is even further reduced.



(a) CORK method

(b) CORK method with low rank

Figure 6.3: Results for the 'gun' problem.

6.6 Conclusions

In this chapter, we have proposed a uniform framework of Compact Rational Krylov (CORK) methods for solving large-scale nonlinear eigenvalue problems. We also introduced a generic but simple representation of structured linearization pencils. These include all the pencils discussed in Chapters 2–5. The family of CORK methods is most applicable in cases where $d \ll n$ and $d \ll m$, with d the degree of the (rational) matrix polynomial, n the original problem dimension, and m the maximum dimension of the subspace.

By representing the subspace $\mathbf{V}$ in a compact form with $\mathbf{V} = (I \otimes Q)\mathbf{U}$, we are able to reduce both the memory cost and the orthogonalization cost. We also proved that the rank of Q, where Q forms an approximation of the eigenspace, is bounded by $m + d$. Therefore, the memory cost reduced from $O(dn \cdot m)$ to $O(n \cdot (d + m))$ and the orthogonalization process only involves $O((d + m)m)$ scalar products of size n instead of $O(dm^2)$. We also briefly discussed locking, purging, and implicit restarting of the CORK method.

The numerical experiments showed that in practice we often get a further reduction when the upper bound $m + d$ on the rank of Q is not attained. This interesting observation appears to be useful for the dynamical approaches where the degree d is not determined before the start of the algorithm.

Chapter 7

More on Applications

Nonlinear eigenvalue problems arise in a lot of different fields of science and engineering. In this chapter, we use the proposed methods of the previous chapters to solve applications from mechanical engineering, quantum physics, and civil engineering. We consider the following three applications which were not solved earlier with the same efficiency and reliability.

1. *Structural dynamics.* Viscoelastic damping material is often used for vibration control and noise reduction. Consequently, the computation of eigenfrequencies and corresponding eigenmodes of these sandwich structures requires the solution of a nonlinear eigenvalue problem.

2. *Computational electronics.* Determining bound states in a semiconductor device with contacts also results in a nonlinear eigenvalue problem. Due to complex square roots, the wanted real eigenvalues are located nearby several branch cuts. Therefore, in order to efficiently compute the bound states and corresponding wave functions, we first subdivide the interval of interest in intervals with only singularities on the endpoints. Next, we apply appropriate transformations to also remove these singularities.

3. *Soil mechanics.* Determining surface waves in a multilayered halfspace requires the solution of a nonlinear eigenvalue problem for every frequency. These nonlinear eigenvalue problems are analytic in the region of interest except for a countable number of isolated singularities. Therefore, we also propose a strategy to remove these singularities.

This chapter is organised as follows. Firstly, we compute eigenfrequencies and corresponding eigenmodes of a clamped sandwich beam in Section 7.1. Next, we determine in Section 7.2 bound states and corresponding wave functions of semiconductor devices with contacts. Finally, we compute dispersion curves of a layered soil in Section 7.3.

7.1 Clamped sandwich beam with viscoelastic core

We consider a symmetric 210 mm long constrained-layer damping (CLD) [80] cantilever beam composed of two cold rolled DC04 steel layers and an ethylene-propylene-diene (EPDM) adhesive core [71], see §1.1.2 and Figure 7.1.

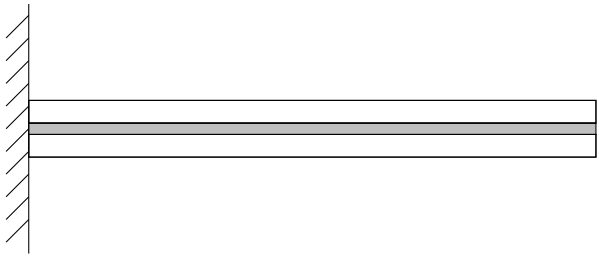


Figure 7.1: Clamped sandwich beam with viscoelastic core.

The beam is discretized using the finite element formulation proposed in [1], into 5 mm long 42 finite elements defined by 43 nodes, which yields 168 degrees of freedom after applying the clamped boundary condition. This results in the following nonlinear eigenvalue problem

$$A(\omega)x = \left(K - \omega^2 M + \frac{G_0 + G_\infty (i\omega\tau)^\alpha}{1 + (i\omega\tau)^\alpha} C\right) x = 0, \tag{7.1}$$

where K, M, and C are constant matrices. The shear modulus of the sandwich beam is described by the four-parameter fractional derivative model, also known as the generalized Zener model. The static shear modulus $G_0 = 350.4\,\text{kPa}$, the asymptotic shear modulus $G_\infty = 3.062\,\text{MPa}$, the relaxation time $\tau = 8.230\,\text{ns}$ and the fractional parameter $\alpha = 0.675$.

In [106] we solved the NLEP (7.1) with the Newton Rational Krylov method. With this method we are able to compute the natural frequencies

$$f_n = \frac{\text{Re}(\omega)}{2\pi},$$

accurately. The 10 smallest eigenvalues of (7.1) and their corresponding relative residuals are given in Table 7.1. Figure 7.2 shows the transversal displacement of the clamped sandwich beam with viscoelastic core in function of x for the 4 smallest natural frequencies.

Table 7.1: The 10 smallest eigenvalues of the ‘sandwich beam’ problem.

ω	$\|A(\omega)\|/\|\omega\|$
1.3089e+02 + 3.9759e+00i	6.6637e-10
7.2337e+02 + 8.2940e+01i	7.1686e-13
1.9207e+03 + 2.9849e+02i	1.3876e-12
3.5800e+03 + 6.5778e+02i	4.9937e-13
5.6749e+03 + 1.1327e+03i	3.0197e-13
8.1832e+03 + 1.7015e+03i	3.5288e-13
1.1097e+04 + 2.3423e+03i	2.0809e-11
1.4415e+04 + 3.0390e+03i	9.2233e-10
1.8141e+04 + 3.7793e+03i	1.4533e-09
2.2280e+04 + 4.5536e+03i	6.6331e-09

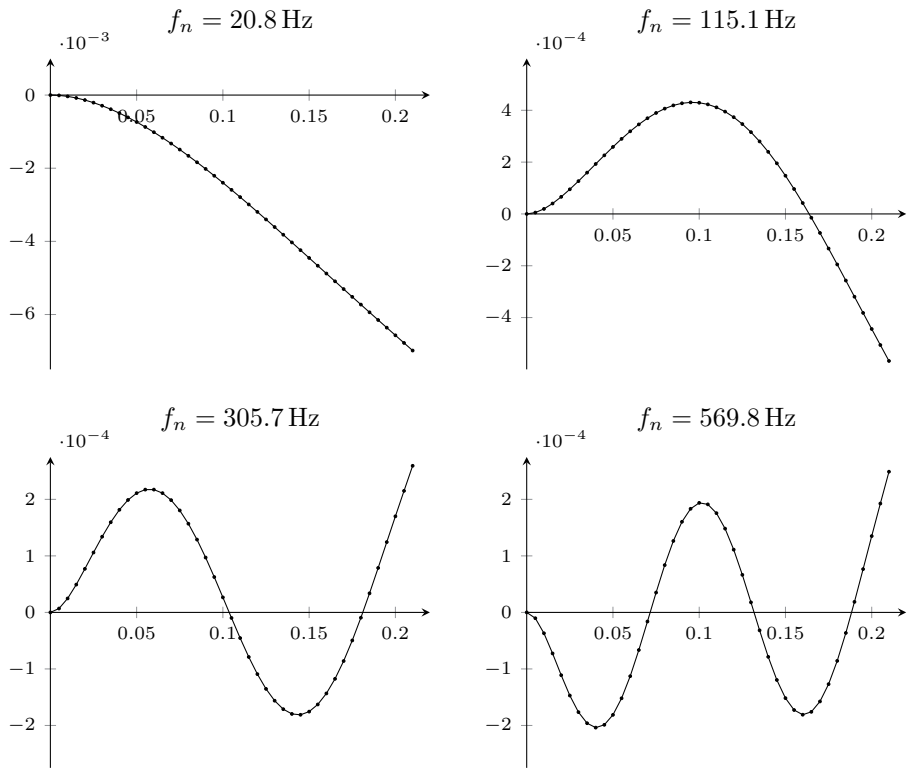


Figure 7.2: Transversal displacement [m] in function of x [m] of the ‘sandwich beam’ problem for the 4 smallest natural frequencies.

7.2 Bound states in semiconductor devices revisited

The computation of bound states in semiconductor devices with contacts gives rise to the following nonlinear eigenvalue problems

$$A(\lambda)x = \left(H - \lambda I + \sum_{\ell=0}^{k} \exp\left(i\sqrt{\lambda - s_\ell} \right) S_\ell \right) x = 0, \tag{7.2}$$

where $\sqrt{\cdot}$ denotes the principal branch of the complex square root, defined by

$$\sqrt{z} = \begin{cases} \sqrt{|z|} \cdot \exp\left(\frac{1}{2} i \arg(z)\right), & z \neq 0 \\ 0, & z = 0 \end{cases},$$

with $\arg(z) \in (-\pi, \pi]$ denoting the argument of the complex number z. Moreover, H and $S_0, \ldots, S_k$ are given constant matrices, and $s_0 < \cdots < s_k$ are given real numbers. The real eigenvalues of (7.2) are the bound states and the corresponding eigenvectors represent the wave functions. Note that, due to the complex square roots, the wanted real eigenvalues of the NLEP (7.2) are located nearby several branch cuts. Therefore, to compute the bound states, we will subdivide the real axis in the interval $[s_{\min}, s_0)$, the intervals $[s_{j-1}, s_j]$ for $j = 1, \ldots, k$, and the interval (s_k, ∞).

A first possibility to deal with these singularities is by rotating the branch cuts in an appropriate way and by using rational approximations. This approach was used in the numerical experiments of §4.3.2. Here we will follow a different approach [109]. We use a holomorphic (also known as analytic) extension of the matrix-valued function $A(\lambda)$ in (7.2) to a neighborhood of the real intervals. Consequently, $A(\lambda)$ can be approximated on the corresponding interval by an interpolating matrix polynomial.

Firstly, we introduce the holomorphic extension in §7.2.1. Next, we compute in §7.2.2 the bound states for a potential describing a particle in a canyon.

7.2.1 Holomorphic extension

We start with the holomorphic extension of $A(\lambda)$ to a neighborhood of the intervals $[s_{j-1}, s_j]$ for $j = 1, \ldots, k$. The remaining intervals $[s_{\min}, s_0)$ and (s_k, ∞) will be discussed further.

A graphical illustration of the holomorphic extension of the interval $[s_{j-1}, s_j]$ is given in Figure 7.3. In this figure, the red dots and lines represent the branch points and branch cuts, respectively. The standard choice for the branch cuts in (7.2) is illustrated in case (a). Next in case (b), the branch cuts corresponding to

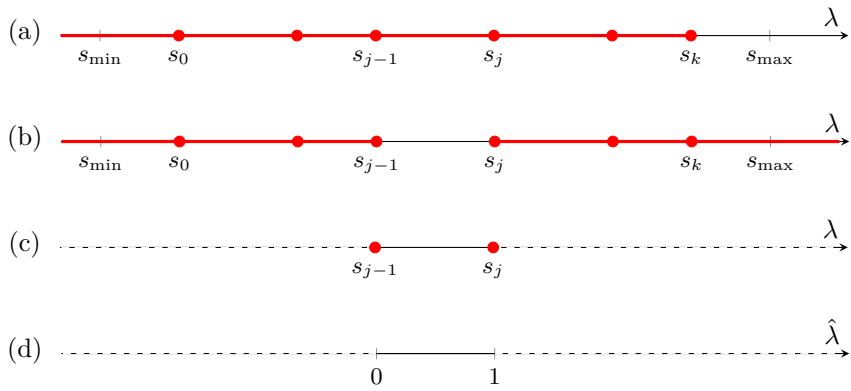


Figure 7.3: Graphical illustration of holomorphic extension.

the branch points $s_j, \ldots, s_k$ are flipped in order to make the interval $[s_{j-1}, s_j]$ branch cut free. Then in case (c), we restrict the eigenvalue computation problem to the subproblem on this interval. Note that it still contains branch cuts on the endpoints. Finally in case (d), we map the interval $[s_{j-1}, s_j]$ to the interval $[0, 1]$ by a transformation from λ to $\hat{\lambda}$ which eliminates the branch points. We now discuss this holomorphic extension in more detail.

For $\lambda \in [s_{j-1}, s_j]$, $A(\lambda)$ in (7.2) can be rewritten as

$$A(\lambda) = H - \lambda I + \sum_{\ell=0}^{j-1} \exp\left(i\sqrt{\lambda - s_\ell}\right) S_\ell + \sum_{\ell=j}^{k} \exp\left(-\sqrt{s_\ell - \lambda}\right) S_\ell, \qquad (7.3)$$

Note that for $\lambda \notin [s_{j-1}, s_j]$ the formulations (7.2) and (7.3) may disagree. But since we are only interested in real eigenvalues, this discrepancy is irrelevant to us. The advantage of formulation (7.3) over the original formulation is that it does not have a branch cut within the open interval (s_{j-1}, s_j). However, the branch points at s_{j-1} and s_j still inhibit a holomorphic extension to a neighborhood of the interval. To eliminate these branch points, we reparameterize the problem. Setting

$$\lambda = \Delta s_j \sin^2\left(\tfrac{\pi}{2}\hat{\lambda}\right) + s_{j-1}, \quad \Delta s_j = s_j - s_{j-1}, \qquad (7.4)$$

we obtain the equivalent nonlinear eigenvalue problem

$$A_j(\hat{\lambda})x = 0, \qquad (7.5)$$

with the eigenvalue parameter $\hat{\lambda} \in [0,1]$ and

$$\begin{aligned} A_j(\hat{\lambda}) = H &- \left(\Delta s_j \sin^2\left(\tfrac{\pi}{2}\hat{\lambda}\right) + s_{j-1}\right) I \\ &+ \sum_{\ell=0}^{j-2} \exp\left(i\sqrt{\Delta s_j \sin^2\left(\tfrac{\pi}{2}\hat{\lambda}\right) + s_{j-1} - s_\ell}\right) S_\ell \\ &+ \exp\left(i\sqrt{\Delta s_j}\sin\left(\tfrac{\pi}{2}\hat{\lambda}\right)\right) S_{j-1} + \exp\left(-\sqrt{\Delta s_j}\cos\left(\tfrac{\pi}{2}\hat{\lambda}\right)\right) S_j \\ &+ \sum_{\ell=j+1}^{k} \exp\left(-\sqrt{s_\ell - s_{j-1} - \Delta s_j \sin^2\left(\tfrac{\pi}{2}\hat{\lambda}\right)}\right) S_\ell. \end{aligned} \tag{7.6}$$

Theorem 7.1. *The nonlinear eigenvalue problems* (7.2) *and* (7.5)–(7.6) *are equivalent in the sense that* $\hat{\lambda} \in [0,1]$ *is an eigenvalue of* (7.5)–(7.6) *if and only if* $\lambda \in [s_{j-1}, s_j]$ *given by* (7.4) *is an eigenvalue of* (7.2).

Proof. Let $\hat{\lambda} \in [0,1]$ be arbitrary and let λ be given by (7.4). Then, we have $\sin\left(\tfrac{\pi}{2}\hat{\lambda}\right), \cos\left(\tfrac{\pi}{2}\hat{\lambda}\right) \in [0,1]$, implying that $\lambda \in [s_{j-1}, s_j]$ and

$$\begin{aligned} \sqrt{\lambda - s_{j-1}} &= \sqrt{\Delta s_j}\sin\left(\tfrac{\pi}{2}\hat{\lambda}\right), \\ \sqrt{s_j - \lambda} &= \sqrt{\Delta s_j\left(1 - \sin^2\left(\tfrac{\pi}{2}\hat{\lambda}\right)\right)} = \sqrt{\Delta s_j}\cos\left(\tfrac{\pi}{2}\hat{\lambda}\right). \end{aligned}$$

Moreover, for $\ell = j, \ldots, k$,

$$i\sqrt{\lambda - s_\ell} = -\sqrt{s_\ell - \lambda}$$

since $\lambda - s_\ell \leq 0$. Combining the above identities yields $A(\lambda) = A_j(\hat{\lambda})$, from which the claim follows. □

To compute eigenvalues in the interval $[s_{\min}, s_0)$, a similar transformation may be employed. To eliminate the branch point at s_0, we substitute

$$\lambda = s_0 - \Delta s_0 \hat{\lambda}^2, \qquad \Delta s_0 = s_0 - s_{\min}, \tag{7.7}$$

leading to the equivalent nonlinear eigenvalue problem

$$A_0(\hat{\lambda})x = 0, \tag{7.8}$$

with the eigenvalue parameter $\hat{\lambda} \in [0,1]$ and

$$\begin{aligned} A_0(\hat{\lambda}) = H &- \left(s_0 - \Delta s_0 \hat{\lambda}^2\right) I + \exp\left(-\sqrt{\Delta s_0}\hat{\lambda}\right) S_0 \\ &+ \sum_{\ell=1}^{k} \exp\left(-\sqrt{s_\ell - s_0 + \Delta s_0 \hat{\lambda}^2}\right) S_\ell. \end{aligned} \tag{7.9}$$

Theorem 7.2. *The nonlinear eigenvalue problems* (7.2) *and* (7.8)–(7.9) *are equivalent in the sense that* $\hat{\lambda} \in [0,1]$ *is an eigenvalue of* (7.8)–(7.9) *if and only if* $\lambda \in [s_{\min}, s_0]$ *given by* (7.7) *is an eigenvalue of* (7.2).

Proof. Let $\hat{\lambda} \in [0,1]$ be arbitrary and let λ be given by (7.7). Then, we can readily verify that $\lambda \in [s_{\min}, s_0]$ and

$$-\sqrt{s_0 - \lambda} = -\sqrt{\Delta s_0}\hat{\lambda}.$$

Moreover, for $\ell = 1, \ldots, k$,

$$i\sqrt{\lambda - s_\ell} = -\sqrt{s_\ell - \lambda}$$

since $\lambda - s_\ell \leq 0$. The proof is completed by noting that these identities imply $A(\lambda) = A_0(\hat{\lambda})$. □

To compute eigenvalues in the interval (s_k, ∞), again a similar transformation may be employed. Before presenting the details, we reduce the unbounded interval (s_k, ∞) to some finite interval $(s_k, s_{\max}]$ using the following estimate.

Theorem 7.3. *Let* $\lambda \geq s_k$ *be a real eigenvalue of the nonlinear eigenvalue problem* (7.2). *Then,* $\lambda \leq s_{\max}$, *where, for any matrix norm* $\|\cdot\|$,

$$s_{\max} = \|H\| + \sum_{\ell=0}^{k} \|S_\ell\|.$$

Proof. By definition of the matrix-valued function A in (7.2), λ is an eigenvalue of (7.2) if and only if it is an eigenvalue of the matrix

$$A_\lambda = H + \sum_{\ell=0}^{k} \exp\left(i\sqrt{\lambda - s_\ell}\right) S_\ell.$$

Moreover, $\lambda \geq s_k$ implies that $\sqrt{\lambda - s_\ell}$ is real for all $\ell = 0, \ldots, k$. Consequently, $\left|\exp\left(i\sqrt{\lambda - s_\ell}\right)\right| = 1$, for $\ell = 0, \ldots, k$, and

$$\lambda \leq \|A_\lambda\| \leq \|H\| + \sum_{\ell=0}^{k} \left|\exp\left(i\sqrt{\lambda - s_\ell}\right)\right| \|S_\ell\| = s_{\max}.$$ □

To eliminate the branch point at s_k, we substitute

$$\lambda = \Delta s_{k+1}\hat{\lambda}^2 + s_k, \qquad \Delta s_{k+1} = s_{\max} - s_k, \tag{7.10}$$

leading to the equivalent nonlinear eigenvalue problem

$$A_{k+1}(\hat{\lambda})x = 0, \tag{7.11}$$

with the eigenvalue parameter $\hat{\lambda} \in [0,1]$ and

$$\begin{aligned} A_{k+1}(\hat{\lambda}) = H &- \left(\Delta s_{k+1}\hat{\lambda}^2 + s_k\right) I \\ &+ \sum_{\ell=0}^{k-1} \exp\left(i\sqrt{\Delta s_{k+1}\hat{\lambda}^2 + s_k - s_\ell}\right) S_\ell \\ &+ \exp\left(i\sqrt{\Delta s_{k+1}}\hat{\lambda}\right) S_k. \end{aligned} \tag{7.12}$$

Theorem 7.4. *The nonlinear eigenvalue problems* (7.2) *and* (7.11)–(7.12) *are equivalent in the sense that* $\hat{\lambda} \in [0,1]$ *is an eigenvalue of* (7.11)–(7.12) *if and only if* $\lambda \in [s_k, s_{\max}]$ *given by* (7.10) *is an eigenvalue of* (7.2).

Proof. Let $\hat{\lambda} \in [0,1]$ be arbitrary and let λ be given by (7.10). Then, one readily verifies that $\lambda \in [s_k, s_{\max}]$ and

$$i\sqrt{\lambda - s_k} = i\sqrt{\Delta s_{k+1}}\hat{\lambda}.$$

The proof is completed by noting that the latter identity implies $A(\lambda) = A_{k+1}(\hat{\lambda})$. □

In this section we showed that the transformed eigenvalue problems (7.5)–(7.6), (7.8)–(7.9), and (7.11)–(7.12) possess holomorphic extensions to a neighborhood of the intervals $[s_{j-1}, s_j]$, $j \in \{1, \ldots, k\}$, $[s_{\min}, s_0]$, and $[s_k, s_{\max}]$, respectively. Thus, we can solve these problems and consequently also the nonlinear eigenvalue problem (7.2) by using for example the Newton Rational Krylov method.

7.2.2 Particle in a canyon problem

We reconsider the `'particle in a canyon'` problem yielding a NLEP of the form (7.2), see §1.1.2. We opt for a potential describing a particle in a canyon

$$U(x,z) = \begin{cases} -U_0\theta(w_1/2 - |z|) & \text{abs}(x) > l/2 \\ -U_0\theta(w_2/2 - |z|) & \text{abs}(x) \le l/2 \end{cases},$$

where U_0 is the depth of the canyon, $\theta(x)$ is the step function and w_1 and w_2 are the width of the contact and the width of the canyon as illustrated in Figure 7.4 (dashed lines). The confinement inside the contacts is stronger than the confinement in the center of the device. This will result in the presence of one or more bound states inside the device.

In [109] we solved the `'particle in a canyon'` problem with the Fully Rational Krylov method for the parameters $x_{\mathrm{R}} = -x_{\mathrm{L}} = 3\,\mathrm{nm}$, $w = 4\,\mathrm{nm}$, $w_1 = 1.6\,\mathrm{nm}$, $w_2 = 2.4\,\mathrm{nm}$, $U_0 = 3\,\mathrm{eV}$, and a particle mass $m = 0.2\,\mathrm{m_e}$. Figure 7.4 shows the wave functions corresponding to the first 5 bound states.

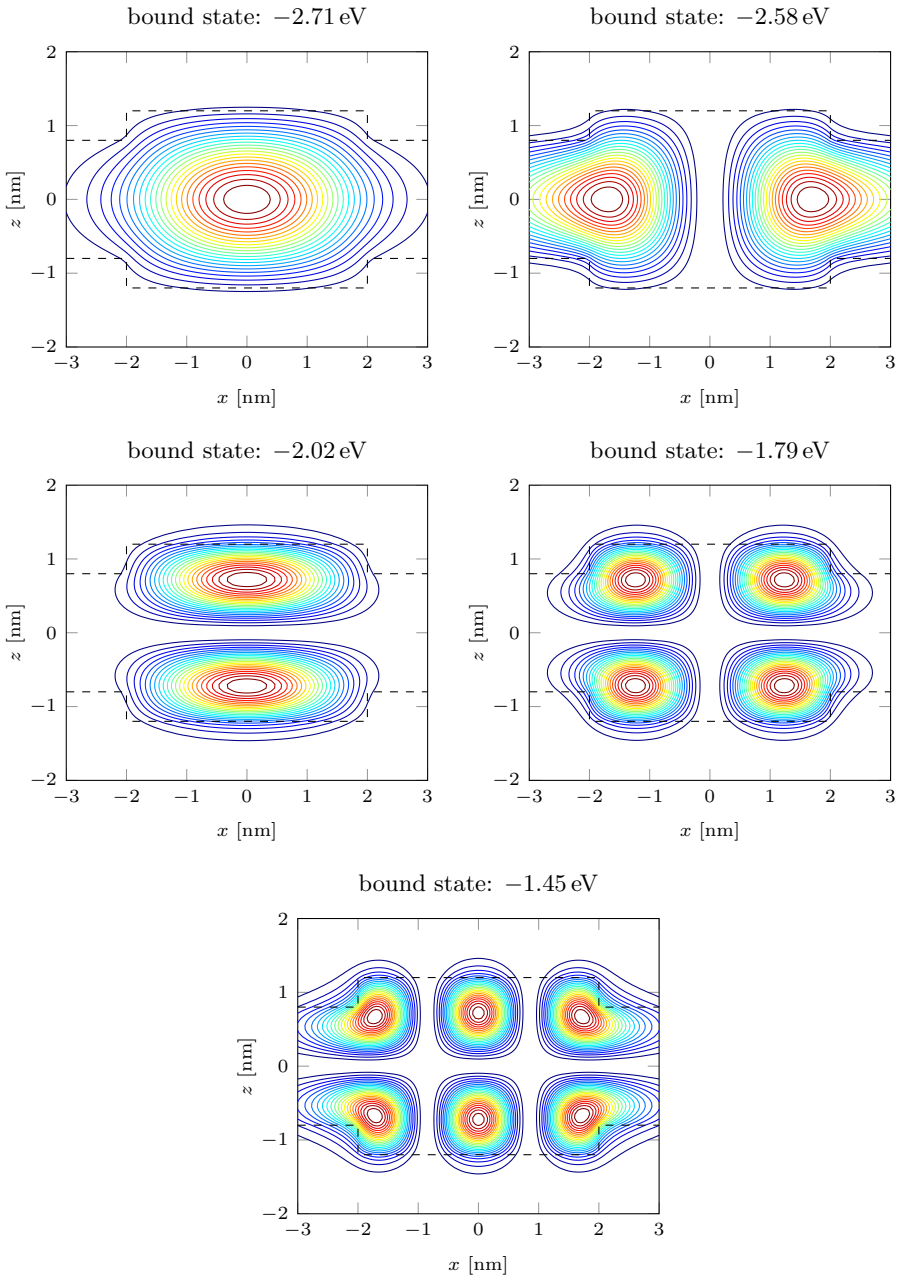


Figure 7.4: Illustration of the canyon potential (dashed line) and contour plot of the squared aptitude of the first 5 bound states in the canyon potential.

7.3 Surface waves in multilayered halfspaces

Determination of surface waves, i.e., free vibrations of a layered medium, requires, for every frequency ω, the solution of the following nonlinear eigenvalue problem in k_x

$$K(k_x, \omega)x = 0, \tag{7.13}$$

where the stiffness matrix-valued function $K(k_x, \omega)$ depends in a nonlinear way on the complex horizontal wavenumber k_x. For more information, see [89, 90, 91].

In several situations, the matrix-valued function $K(k_x, \omega)$ contains a few poles inside the target set Σ. Thus, $K(k_x, \omega)$ is analytic on Σ except for a countable number of isolated singularities. In this situation a rational approximation of $K(k_x, \omega)$ can be used. However, the use of an analytic function is preferable to a rational approximation with poles inside the target set, because we then know that the maximum error is attained on the boundary. Therefore, we will firstly remove these poles. A brute-force approach consists of multiplying $K(k_x, \omega)$ by a scalar polynomial which has the poles of $K(k_x, \omega)$ as roots. Although we obtain matrix-valued functions which are analytic on the target set, this approach might convert the removed poles into eigenvalues with possibly high multiplicities.

Firstly, we describe in §7.3.1 how poles can be removed without converting into eigenvalues. Next, we compute the dispersion and attenuation curves of a layered halfspace in §7.3.2.

7.3.1 Removing poles

Suppose that the matrix-valued function $A(\lambda)$ is analytic except for a single point $\xi \in \mathbb{C}$. We assume that this singularity is a pole of multiplicity one. If we multiply $A(\lambda)$ from the left by $\lambda - \xi$,

$$\widetilde{A}(\lambda) := (\lambda - \xi)A(\lambda),$$

is a holomorphic extension of $A(\lambda)$. However, this approach might introduce an extra eigenvalue at the location of the pole ξ, since

$$\det \widetilde{A}(\lambda) = (\lambda - \xi)^n \det A(\lambda).$$

The multiplicity of this extra eigenvalue $\lambda = \xi$ can be large but at most $n - 1$. To see this, we consider the Laurent series of $A(\lambda)$ about the pole ξ

$$A(\lambda) = \frac{\operatorname{Res} A(\xi)}{\lambda - \xi} + \sum_{i=0}^{\infty} C_i(\lambda - \xi)^i, \tag{7.14}$$

where $\operatorname{Res} A(\xi)$ is the residue of $A(\lambda)$ at the pole ξ and $C_i \in \mathbb{C}^{n\times n}$. Next, multiplying (7.14) from the left by $\lambda - \xi$, yields

$$\widetilde{A}(\lambda) = \operatorname{Res} A(\xi) + \sum_{i=0}^{\infty} C_i(\lambda - \xi)^{i+1}. \tag{7.15}$$

Note that in case the residue $\operatorname{Res} A(\xi)$ is of full rank, no extra eigenvalue at ξ is introduced. However, in case $r < n$, with r the rank of $\operatorname{Res} A(\xi)$, an extra eigenvalue with multiplicity $n - r$ is introduced at ξ.

In order to avoid the introduction of an extra eigenvalue at ξ, we consider the singular value decomposition of the residue of $A(\xi)$

$$\operatorname{Res} A(\xi) =: USV^*,$$

where U and V are unitary matrices and S is a diagonal matrix of rank r. Next, we have

$$U^* \operatorname{Res} A(\xi) = SV^* = \left[\begin{array}{c} s_1 v_1^* \\ \vdots \\ \hline s_r v_r^* \\ \hline 0_{(n-r)\times n} \end{array}\right],$$

where $s_1, \ldots, s_r$ are the singular values and $v_1, \ldots, v_r$ the right singular vectors. Consequently, multiplying (7.14) from the right by U^*, yields

$$U^* A(\lambda) = \frac{SV^*}{\lambda - \xi} + \sum_{i=0}^{\infty} U^* C_i(\lambda - \xi)^i. \tag{7.16}$$

Now, instead of multiplying every row of (7.14) by $\lambda - \xi$ as in (7.15), we only multiply the first r rows of (7.16) by $\lambda - \xi$ and define $\widehat{A}(\lambda)$ as follows

$$\begin{aligned} \widehat{A}(\lambda) &= \begin{bmatrix} (\lambda - \xi) I_r & \\ & I_{n-r} \end{bmatrix} U^* A(\lambda) \\ &= SV^* + \sum_{i=0}^{\infty} \begin{bmatrix} (\lambda - \xi)^{i+1} I_r & \\ & (\lambda - \xi)^i I_{n-r} \end{bmatrix} U^* C_i. \end{aligned}$$

Hence, we obtain an analytic function without introducing an extra eigenvalue at ξ.

We now generalize this approach to the case $A(\lambda)$ is analytic on a target set except for a countable number of poles $\xi_1, \ldots, \xi_k$. Therefore, we will recursively repeat the procedure described higher for every pole ξ_i.

Let $A_0(\lambda) := A(\lambda)$ and for $i = 1, \ldots, k$, we define

$$A_i(\lambda) := \Big(D_i(\lambda) U_i^*\Big) A_{i-1}(\lambda),$$

where U_i is obtained from the singular value decomposition of

$$\operatorname{Res} A_{i-1}(\xi_i) =: U_i S_i V_i^*, \tag{7.17}$$

and $D_i(\lambda)$ defined as follows

$$D_i(\lambda) := \begin{bmatrix} (\lambda - \xi_i) I_{r_i} & \\ & I_{n-r_i} \end{bmatrix},$$

with $r_i = \operatorname{rank}(S_i)$. Note that the residue (7.17) can be expressed in terms of the residue of the original matrix-valued function $A(\lambda)$ at the same pole

$$\operatorname{Res} A_{i-1}(\xi_i) = \Big(D_{i-1}(\xi_i) U_{i-1}^*\Big) \cdots \Big(D_2(\xi_i) U_2^*\Big)\Big(D_1(\xi_i) U_1^*\Big) \operatorname{Res} A(\xi_i).$$

The matrix-valued function

$$\widehat{A}(\lambda) := A_k(\lambda) = \Big(D_k(\lambda) U_k^*\Big) \cdots \Big(D_2(\lambda) U_2^*\Big)\Big(D_1(\lambda) U_1^*\Big) A(\lambda),$$

is analytic on the target set and does not introduce extra eigenvalues at the poles $\xi_1, \ldots, \xi_k$. Thus, we can compute the eigenvalues of $A(\lambda)$ in the target set as solutions of the nonlinear eigenvalue problem

$$\widehat{A}(\lambda) x = 0,$$

by using for example a Compact Rational Krylov method.

7.3.2 Surface waves problem

We consider the `'surface waves'` problem, see §1.1.2, for a layered soil consisting of two homogeneous layers with a thickness of 5 m on a homogeneous halfspace. The soil profile is described in Table 7.2 with h the layer thickness, C_s the shear wave velocity, C_p the dilatational wave velocity, D_s the damping ratio for the shear waves, D_p the damping ratio for the dilatational waves, and ρ the soil density.

Table 7.2: Soil properties.

Layer	h [m]	C_s [m/s]	C_p [m/s]	D_s [–]	D_p [–]	ρ [kg/m^3]
1	5	150	300	0.03	0.03	1800
2	5	250	500	0.03	0.03	1800
3	∞	400	900	0.03	0.03	1800

The aim here is to compute the dispersion curves $C_R(\omega) = \omega/\mathrm{Re}(k_x)$ and attenuation curves $A_R(\omega) = -\mathrm{Im}(k_x)$ of all in-plane surface waves or Rayleigh waves in a frequency range up to 100 Hz. Therefore, we need to compute the horizontal wavenumbers k_x for each frequency, yielding 6-dimensional nonlinear eigenvalue problems of the form (7.13). The target set Σ, a disk in the complex plane with centre 2 and radius 1, is the same for all frequencies f if we express these nonlinear eigenvalue problems in terms of the dimensionless horizontal wavenumber $\bar{k}_x$

$$\bar{k}_x = \frac{C_s^{(3)}}{\omega} k_x,$$

where $C_s^{(3)}$ is the shear wave velocity of the halfspace and $\omega = 2\pi f$. For small frequencies ($f < 20$ Hz) the matrix-valued function $K(k_x, \omega)$ is analytic on the target set Σ. However, for higher frequencies $K(k_x, \omega)$ contains a few poles inside Σ.

We now illustrate the computation of the horizontal wavenumbers k_x for the frequency $f = 50$ Hz. Firstly, we applied `ratdisk` [39] to each element of the matrix-valued function $K(k_x, 100\pi)$ yielding the poles, with corresponding residues, inside and close to the target set. We obtained 9 poles, indicated in Figure 7.5 by "•", of which 5 are lying inside the target set Σ. Secondly, we removed these poles, as explained in §7.3.1, yielding a matrix-valued function $\widehat{K}(k_x)$ which is analytic on, and close to, the target set. Thirdly, we constructed

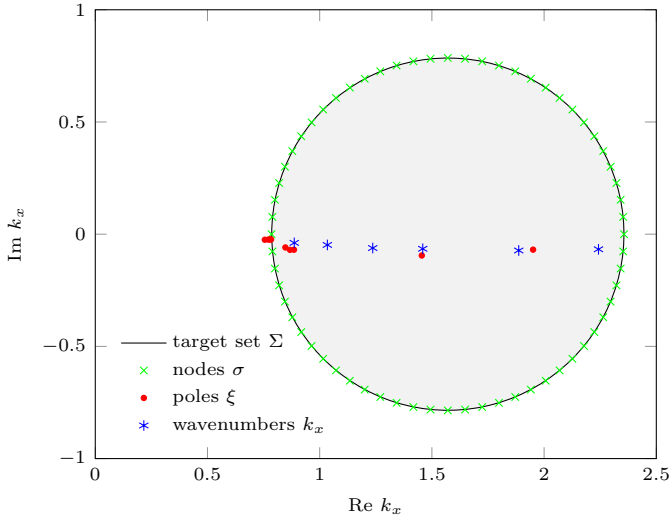
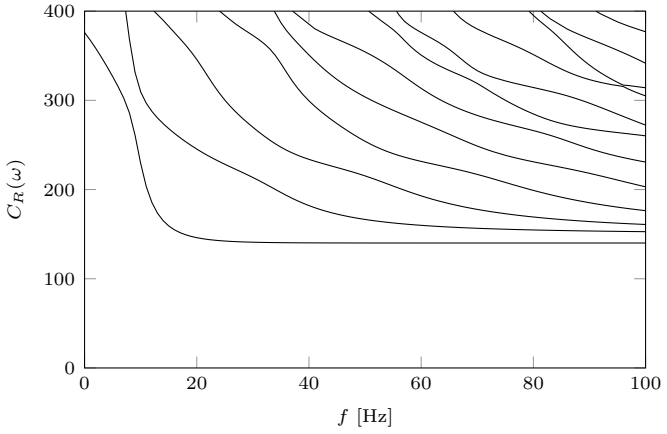


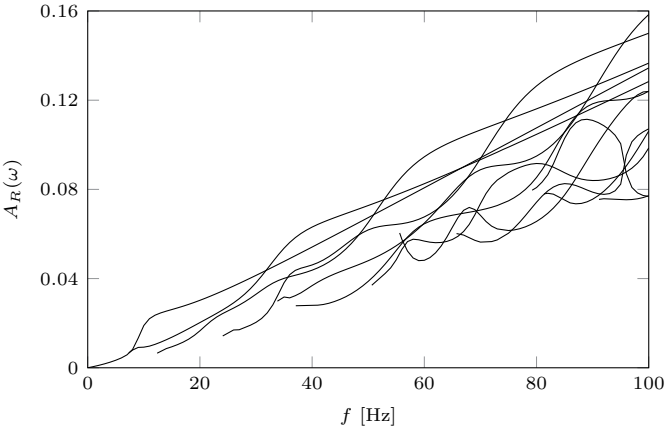
Figure 7.5: `'Surface waves'` problem for $f = 50$ Hz.

a matrix polynomial $P(k_x)$ in barycentric Lagrange form, by interpolating $\widehat{K}(k_x)$ in 64 equidistant interpolation nodes σ, indicated in Figure 7.5 by "$\times$". Finally, we computed the horizontal wavenumbers k_x with the Compact Rational Krylov method applied to the linearization pencil of $P(k_x)$ obtained by Theorem 2.8.

By applying this approach for all frequencies $f = 1, 2, \ldots, 100\,\text{Hz}$, we determined the surface waves in the frequency range up to 100 Hz. Figure 7.6 shows the dispersion curves $C_R(\omega)$ and attenuation curves $A_R(\omega)$ of the first 12 Rayleigh waves.



(a) phase velocity



(b) attenuation coefficient

Figure 7.6: `'Surface waves'` problem: (a) dispersion curves and (b) attenuation curves.

Chapter 8

Conclusion

The global aim of this dissertation has been the development of efficient, reliable, and globally convergent rational Krylov methods for solving both small-scale and large-scale nonlinear eigenvalue problems. The key ingredients of these methods are linearization of matrix polynomials (Chapter 2), dynamic polynomial interpolation (Chapter 3) and rational interpolation (Chapter 4). A connection with the operator setting is also made (Chapter 5).

A new uniform framework for representing linearization pencils is proposed, resulting in Compact Rational Krylov (CORK) methods which maximally exploit the structure of the linearization pencils in the rational Krylov method (Chapter 6). This is all combined in a state-of-the-art publicly available numerical algorithm for solving nonlinear eigenvalue problems. Linearization significantly increases both the memory cost and the orthogonalization cost. However, for large-scale problems with moderate degree of the (rational) matrix polynomial, the memory and orthogonalization costs in the CORK algorithm are of the same order of magnitude as for solving a linear eigenvalue problem of the same dimension.

Finally, the CORK algorithm is applied to some real applications originating from vibration control and noise reduction of sandwich structures, electronic transport in semiconductor devices, and spectral analysis of layered soils (Chapter 7).

This chapter is organized as follows. Firstly, we give a chapter by chapter overview of the contributions in Section 8.1. Secondly, we formulate a few possible directions for future research in Section 8.2.

8.1 Contributions

In this thesis, we developed new rational Krylov methods for solving nonlinear eigenvalue problems. We now give a chapter by chapter overview of our original contributions:

Chapter 2:

- *Strong linearizations for matrix polynomials in Lagrange basis.* Strong linearizations have the property that there is a one-to-one correspondence between the eigenpairs of a matrix polynomial and the eigenpairs of the linearization pencil. We introduced a strong linearization for matrix polynomials in modified or barycentric Lagrange form. This linearization is also generalized to Hermite Lagrange and barycentric Hermite matrix polynomials.

Chapter 3:

- *Dynamic polynomial interpolation in Newton basis.* Interpolation in Newton basis has the advantage that adding an extra interpolation point does not change the previous Newton basis functions and their coefficients. Consequently, adding a new interpolation point does not affect the previous linearization pencil and only adds one block row and column. Therefore, using interpolating matrix polynomials in Newton basis for approximating the matrix-valued function of the nonlinear eigenvalue problem results in a better approximation than a truncated Taylor series of the same degree and yields a more global approximation on the target set.

- *Newton Rational Krylov method.* The combination of dynamical polynomial interpolation in Newton basis, linearization, and the rational Krylov method is a promising technique for solving nonlinear eigenvalue problems. We introduced the Newton Rational Krylov method which is started with a short starting vector and where the shifts are no free parameters but equal to the interpolation points. This yields a dynamic algorithm with a growing pencil in every iteration and where neither the degree of the interpolating polynomial, nor the interpolation points need to be fixed in advance.

- *Global eigenvalue search.* The use of different interpolation points in combination with Hermite interpolation in the Newton Rational Krylov method makes the algorithm very suitable for a global eigenvalue search.

- *Local eigenvalue correction.* The freedom of choosing in every iteration of the Newton Rational Krylov method the next interpolation point / shift based on the eigenvalue approximation makes the algorithm also suitable for local eigenvalue correction. We illustrated that by using Ritz values as next interpolation points, the Newton Rational Krylov method converges in less iterations than Newton's method.

- *Low rank exploitation.* The nonlinear eigenvalue problem consists in several applications of a polynomial part and a nonlinear part which is of low rank. In this situation, the low rank structure can be exploited by adapting the linearization pencil in such a way that its dimension is significantly reduced. Consequently, less memory is required to store the subspace and also the orthogonalization cost is significantly reduced.

Chapter 4:

- *Linear rational interpolation in Newton basis.* Polynomial interpolation works very well if the interpolating function is an entire function or when its singularities are sufficiently far away from the target set. However, in the proximity of singularities and branch cuts, rational interpolation is indispensable. We generalized polynomial Newton interpolation to linear rational interpolation in Newton basis and proposed its corresponding linearization which has similar properties as the polynomial one.

- *Rational approximation of the nonlinearity.* The convergence of methods that rely on polynomial interpolation of the matrix-valued function of the nonlinear eigenvalue problem is limited by the convergence of polynomial approximations. However, in case these matrix-valued functions are difficult to approximate by polynomials, the performance of the rational Krylov methods will be limited by the accuracy of the underlying polynomial expansion. Therefore, we introduced the use of linear rational interpolating polynomials for solving nonlinear eigenvalue problems.

- *Fully Rational Krylov method.* The combination of linear rational interpolation in Newton basis, linearization, and the rational Krylov method has led, in collaboration with Stefan Güttel (The University of Manchester), to the introduction of the Fully Rational Krylov method. This method has 3 different variants: a static, a dynamic, and a hybrid variant. All these variants rely on building an accurate rational approximation of the matrix-valued function of the nonlinear eigenvalue problem over the whole target set.

- *Optimal choice of interpolation nodes and poles.* The choice of the interpolation nodes and poles has a major impact on the convergence of

the Fully Rational Krylov method. We proposed the Leja-Bagby points as an asymptotically optimal strategy for achieving uniform fast convergence of the rational approximation on the target set. However, this choice is not always advantageous for the rational Krylov method to quickly find all the eigenvalues inside the target set. Therefore, the hybrid variant of the Fully Rational Krylov method truncates the converged rational expansion such that the next shifts can freely be chosen.

- *Scaling of the rational basis functions.* The scaling of the rational basis functions can remarkably improve both the stability and the convergence of the Fully Rational Krylov method, although scaling has no effect on the eigenvalues of the linearization pencil. We proposed a scaling procedure such that the evaluation of the basis functions during the sampling is not affected by numerical overflow or underflow.

- *Publicly available software.* The Fully Rational Krylov algorithm has been implemented in MATLAB. The matrix-valued function in a nonlinear eigenvalue problem can be represented in different ways. A first possibility is a function handle which returns a matrix for a given scalar value. A second way is by using the property that every matrix-valued function can be written as a sum of constant matrices times scalar nonlinear functions. We make here a distinction between the polynomial part and the non-polynomial part of the nonlinear matrix-valued function. Finally, if applicable, the low rank structure of the nonlinear part is taken into account. The software is publicly available on `http://twr.cs.kuleuven.be/research/software/nleps/nleigs.html`.

Chapter 5:

- *Low rank exploitation in the operator setting.* In collaboration with Elias Jarlebring, we generalized the low rank exploitation, introduced for the Newton Rational Krylov method, to the operator setting. The rank-exploiting Infinite Arnoldi method yields a significant memory and orthogonalization cost reduction compared to the standard Infinite Arnoldi method.

Chapter 6:

- *Uniform representation of linearization pencils.* A wide range of linearization pencils, such as the ones studied in Chapters 2–5, exhibit a similar block and Kronecker structure. We introduced a generic but simple representation of these structured linearization pencils. Furthermore, their eigenvectors also have a Kronecker structure which is related to the eigenvectors of the corresponding (rational) matrix polynomial.

- *Compact rational Krylov decomposition.* The major difficulty of rational Krylov methods using linearizations for large n is the growing memory and orthogonalization costs with the iteration count. In general, they are proportional to the degree of the polynomial, i.e., at iteration j, the storage cost of the iteration vectors is of order $d \cdot j$ vectors of size n and the orthogonalization cost is of order $d \cdot j^2$ scalar products of size n. We introduced the compact rational Krylov decomposition such that the memory cost is only of order $d + j$ vectors of size n and the orthogonalization cost is of order $(d + j)j$ scalar products of size n.

- *Compact Rational Krylov method.* The combination of a uniform representation of linearization pencils and the compact rational Krylov decomposition has led to the introduction of the new framework of compact rational Krylov methods for solving nonlinear eigenvalue problems. The state-of-the-art Compact Rational Krylov (CORK) algorithm exploits structure as much as possible such that for large n and $d \ll n$ the memory and orthogonalization costs are of the same order of magnitude as for solving a linear eigenvalue problem of the same dimension.

- *Locking and restarting in a natural way.* Since the CORK method is a special variant of the rational Krylov method, we can also perform implicit restarting in order to keep the memory and orthogonalization costs under control. Furthermore, locking of converged Ritz values and deflation of unwanted ones can be done in a natural and numerical stable way.

- *Publicly available software.* With the CORK algorithm we introduced a state-of-the-art numerical algorithm for solving nonlinear eigenvalue problems, both small- and large-scale. By using an appropriate representation of the nonlinear eigenvalue problem, the storage of the linearization pencils only requires a minor extra memory cost. Furthermore, by exploitation of this structure only linear algebra operations on matrices and vectors of the original nonlinear problem's dimension are involved. Finally, by using the compact representation of the subspace the memory and orthogonalization costs are reduced to a minimum. The CORK algorithm has been implemented in MATLAB and is publicly available on `http://twr.cs.kuleuven.be/research/software/nleps/cork.html`.

Chapter 7:

- *Application in structural dynamics: constrainted-layer damping.* The use of viscoelastic damping material in sandwich structures results in nonlinear eigenvalue problems for vibration control and noise reduction. In collaboration with Natalia Navarrete (KU Leuven, Department of Mechanical

Engineering), we solved nonlinear eigenvalue problems originating from these viscoelastic damping technologies.

- *Application in computational electronics: bound states.* The determination of bound states in semiconductor devices with contacts also requires the solution of a nonlinear eigenvalue problem. In collaboration with William Vandenberghe (IMEC, University of Texas at Dallas) and Cedric Effenberger (EPFL), we developed an efficient numerical method, based on holomorphic extension, for computing these bound states. The key idea here is applying transformations in the complex plane to the nonlinear eigenvalue problem, such that branch-cuts are moved away from undesirable locations.

- *Application in soil mechanics: surface waves.* The spectral analysis of a layered soil needs the solution of a nonlinear eigenvalue problem for several frequencies. In collaboration with Mattias Schevenels (KU Leuven, Department of Architecture), we are able to solve these nonlinear eigenvalue problems efficiently.

Not reported in this thesis

- *Computing the distance to instability.* A linear time-invariant continuous dynamical system is stable if all eigenvalues lie strictly in the left half of the complex plane. However, this is not a robust measure because stability is no longer guaranteed when the system parameters are perturbed. Therefore, the distance to instability is used as a robust measure for stability against perturbations. In collaboration with Dries Verhees (Ma student KU Leuven) and Nicola Guglielmi (University of L'Aquila), we developed two classes of fast algorithms for computing the distance to instability of large-scale nonlinear eigenvalue problems [110].

- *Computing the pseudospectral abscissa.* In collaboration with Emre Mengi (Koç University), we prepare work on the computation of pseudospectral abscissa of large-scale nonlinear eigenvalue problems. This problem is turned into an eigenvalue optimization problem, for which a novel algorithm guaranteeing convergence to the globally rightmost point of the pseudospectrum is proposed.

8.2 Future research

We discuss a few possible directions for future research:

- *Balancing of linearization pencils for Krylov methods.* Although balancing has no influence on the eigenvalues in exact arithmetic, it can significantly improve the eigenvalue accuracy in numerical algorithms, e.g., diagonal scaling for dense eigenvalue solvers. Furthermore, balancing of linearization pencils affects the corresponding basis functions and therefore also the scalar product in Krylov methods. Hence, the convergence of these methods can be improved by an appropriate balancing. Moreover, since iterative methods only compute a subset of the eigenvalues, a balancing technique with respect to only the eigenvalues inside a given target set is also required.

- *Convergence analysis of low rank exploitation.* By exploiting low rank structure of the nonlinear part in nonlinear eigenvalue problems, we obtained linearization pencils of much smaller dimensions. By this the scalar product in Krylov methods is changed compared to the standard no low rank exploiting linearization pencils. In practice, it turns out that the use of low rank exploiting linearizations has a positive effect on the eigenvalue convergence. However, more theoretical convergence analysis will be required.

- *Contour integration via rational Krylov.* The Cauchy integral reformulation of the nonlinear eigenvalue problem, used in contour integral methods, can be interpreted as a rational filter [103]. Furthermore, an implicit filtering by rational functions can be efficiently performed for the rational Krylov method [29]. Therefore, contour integration via rational Krylov for solving the nonlinear eigenvalue problem should result in computationally more efficient methods than the current contour integration methods. Furthermore, locking of converged eigenvalues can in the rational Krylov method be performed in a robust way.

- *Software package for solving nonlinear eigenvalue problems.* The integration of the different software components in one general open source software package and writing a software oriented paper, including details on the implementation aspects, is ongoing.

Bibliography

[1] K. Amichi and N. Atalla, *A new 3D finite element for sandwich beams with a viscoelastic core*, J. Vib. Acoust., 131 (2009), 021010. (Cited on p. 138)

[2] A. Amiraslani, *New Algorithms for Matrices, Polynomials and Matrix Polynomials*, PhD thesis, University of Western Ontario, London, Ontario, 2006. (Cited on p. 30)

[3] A. Amiraslani, R. M. Corless, and P. Lancaster, *Linearization of matrix polynomials expressed in polynomial bases*, IMA J. Numer. Anal., 29 (2009), pp. 141–157. (Cited on pp. 4, 24, 30, 35, 38, 116, and 119)

[4] P. M. Anselone and L. B. Rall, *The solution of characteristic value-vector problems by Newton's method*, Numer. Math., 11 (1968), pp. 38–45. (Cited on p. 9)

[5] E. N. Antoniou and S. Vologiannidis, *A new family of companion forms of polynomial matrices*, Electron. J. Linear Algebra, 11 (2004), pp. 78–87. (Cited on p. 116)

[6] W. E. Arnoldi, *The principle of minimized iterations in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29. (Cited on p. 11)

[7] J. Asakura, T. Sakurai, H. Tadano, T. Ikegami, and K. Kimura, *A numerical method for nonlinear eigenvalue problems using contour integrals*, JSIAM Lett., 1 (2009), pp. 52–55. (Cited on p. 9)

[8] T. Bagby, *The modulus of a plane condenser*, J. Math. Mech., 17 (1967), pp. 315–329. (Cited on p. 76)

[9] T. Bagby, *On interpolation by rational functions*, Duke Math. J., 36 (1969), pp. 95–104. (Cited on pp. 68, 75, and 76)

[10] J. Baglama, D. Calvetti, and L. Reichel, *Fast Leja points*, Electron. Trans. Numer. Anal., 7 (1998), pp. 124–140. (Cited on pp. 58 and 67)

[11] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. A. Van der Vorst, eds., *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2000. (Cited on pp. 15 and 17)

[12] Z. Bai and Y. Su, *SOAR: A second-order Arnoldi method for the solution of the quadratic eigenvalue problem*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 640–659. (Cited on pp. 3 and 123)

[13] J.-P. Berrut and L. N. Trefethen, *Barycentric Lagrange interpolation*, SIAM Rev., 46 (2004), pp. 501–517. (Cited on pp. 26 and 27)

[14] T. Betcke, N. J. Higham, V. Mehrmann, C. Schröder, and F. Tisseur, *NLEVP: A collection of nonlinear eigenvalue problems*, ACM Trans. Math. Software, 39 (2013), pp. 7:1–7:28. (Cited on pp. 8, 58, 61, 81, and 134)

[15] T. Betcke and H. Voss, *A Jacobi–Davidson-type projection method for nonlinear eigenvalue problems*, Futur. Gener. Comput. Syst., 20 (2004), pp. 363–372. (Cited on p. 9)

[16] W.-J. Beyn, *An integral method for solving nonlinear eigenvalue problems*, Linear Algebra Appl., 436 (2012), pp. 3839–3863. (Cited on p. 9)

[17] J. P. Boyd, *Finding the zeros of a univariate equation: Proxy rootfinders, Chebyshev interpolation, and the companion matrix*, SIAM Rev., 55 (2013), pp. 375–396. (Cited on p. 66)

[18] D. Breda, *Solution operator approximations for characteristic roots of delay differential equations*, Appl. Numer. Math., 56 (2006), pp. 305–317. (Cited on p. 104)

[19] D. Breda, S. Maset, and R. Vermiglio, *Pseudospectral differencing methods for characteristic roots of delay differential equations*, SIAM J. Sci. Comput., 27 (2005), pp. 482–495. (Cited on pp. 103 and 104)

[20] D. Breda, S. Maset, and R. Vermiglio, *Pseudospectral approximation of eigenvalues of derivative operators with non-local boundary conditions*, Appl. Numer. Math., 56 (2006), pp. 318–331. (Cited on p. 104)

[21] D. Breda, S. Maset, and R. Vermiglio, *Numerical approximation of characteristic values of partial retarded functional differential equations*, Numer. Math., 113 (2009), pp. 181–242. (Cited on p. 104)

[22] D. Breda, S. Maset, and R. Vermiglio, *TRACE-DDE: a Tool for Robust Analysis and Characteristic Equations for Delay Differential Equations*, in Topics in Time Delay Systems, J. J. Loiseau, W. Michiels, S.-I. Niculescu, and R. Sipahi, eds., vol. 388 of Lecture Notes in Control and Information Sciences, Springer Berlin Heidelberg, Berlin, 2009, pp. 145–155. (Cited on p. 104)

[23] J. C. Butcher, R. M. Corless, L. Gonzalez-Vega, and A. Shakoori, *Polynomial algebra for Birkhoff interpolants*, Numer. Algorithms, 56 (2011), pp. 319–347. (Cited on p. 29)

[24] R. M. Corless, *Generalized companion matrices in the Lagrange basis*, in Proceedings EACA, L. González Vega and T. Recio, eds., Santander, Spain, June 2004, Universidad de Cantabria, pp. 317–322. (Cited on p. 30)

[25] R. M. Corless, A. Shakoori, D. A. Aruliah, and L. Gonzalez-Vega, *Barycentric Hermite interpolants for event location in initial-value problems*, JNAIAM. J. Numer. Anal. Ind. Appl. Math., 3 (2008), pp. 1–16. (Cited on p. 33)

[26] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart, *Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization*, Math. Comp., 30 (1976), pp. 772–795. (Cited on p. 12)

[27] P. J. Davis, *Interpolation and Approximation*, Dover books on advanced mathematics, Blaisdell Publishing Company, New York, NY, 1963. (Cited on p. 33)

[28] C. de Boor, *Divided differences*, Surv. Approx. Theory, 1 (2005), pp. 46–69. (Cited on p. 36)

[29] G. De Samblanx, K. Meerbergen, and A. Bultheel, *The implicit application of a rational filter in the RKS method*, BIT, 37 (1997), pp. 925–947. (Cited on pp. 17, 19, 128, and 157)

[30] A. Edrei, *Sur les déterminants récurrents et les singularités d'une fonction donnée par son développement de Taylor*, Compos. Math., 7 (1940), pp. 20–88. (Cited on p. 58)

[31] C. Effenberger, *Robust Solution Methods for Nonlinear Eigenvalue Problems*, PhD thesis, EPFL, Lausanne, 2013. (Cited on p. 9)

[32] C. Effenberger, *Robust successive computation of eigenpairs for nonlinear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 1231–1256. (Cited on p. 9)

[33] C. Effenberger and D. Kressner, *Chebyshev interpolation for nonlinear eigenvalue problems*, BIT, 52 (2012), pp. 933–951. (Cited on pp. 9 and 119)

[34] M. Fiedler, *A note on companion matrices*, Linear Algebra Appl., 372 (2003), pp. 325–331. (Cited on p. 116)

[35] J. G. F. Francis, *The QR transformation a unitary analogue to the LR transformation—part 1*, Comput. J., 4 (1961), pp. 265–271. (Cited on p. 3)

[36] J. G. F. Francis, *The QR transformation—part 2*, Comput. J., 4 (1962), pp. 332–345. (Cited on p. 3)

[37] I. Gohberg, M. A. Kaashoek, and P. Lancaster, *General theory of regular matrix polynomials and band Toeplitz operators*, Integral Equations Operator Theory, 11 (1988), pp. 776–882. (Cited on pp. 24 and 25)

[38] I. Gohberg, P. Lancaster, and L. Rodman, *Matrix Polynomials*, Academic Press, New York, 1982. (Cited on pp. 4 and 24)

[39] P. Gonnet, R. Pachón, and L. N. Trefethen, *Robust rational interpolation and least-squares*, Electron. Trans. Numer. Anal., 38 (2011), pp. 146–167. (Cited on p. 149)

[40] A. A. Gončar, *Zolotarev problems connected with rational functions*, Math. USSR Sb., 7 (1969), pp. 623–635. (Cited on p. 76)

[41] M. Gugat, *Boundary feedback stabilization by time delay for one-dimensional wave equations*, IMA J. Math. Control Inform., 27 (2010), pp. 189–203. (Cited on p. 110)

[42] S. Güttel, *Rational Krylov Methods for Operator Functions*, PhD thesis, TU Bergakademie Freiberg, Freiberg, 2010. (Cited on p. 72)

[43] S. Güttel, *Rational Krylov approximation of matrix functions: Numerical methods and optimal pole selection*, GAMM-Mitt., 36 (2013), pp. 8–31. (Cited on pp. 72 and 77)

[44] S. Güttel, R. Van Beeumen, K. Meerbergen, and W. Michiels, *NLEIGS: A class of fully rational Krylov methods for nonlinear eigenvalue problems*, SIAM J. Sci. Comput., 36 (2014), pp. A2842–A2864. (Cited on pp. 20, 56, 69, 72, 75, 78, 80, 116, and 119)

[45] S. Hammarling, C. J. Munro, and F. Tisseur, *An algorithm for the complete solution of quadratic eigenvalue problems*, ACM Trans. Math. Software, 39 (2013), pp. 18:1–18:19. (Cited on p. 3)

[46] N. J. Higham, *The numerical stability of barycentric Lagrange interpolation*, IMA J. Numer. Anal., 24 (2004), pp. 547–556. (Cited on pp. 27 and 28)

[47] N. J. Higham, *Functions of Matrices: Theory and Computation*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2008. (Cited on p. 36)

[48] H. Igarashi, Y. Sugawara, and T. Honma, *A numerical computation of external Q of resonant cavities*, IEEE Trans. Magn., 31 (1995), pp. 1642–1645. (Cited on p. 8)

[49] E. Jarlebring, K. Meerbergen, and W. Michiels, *A Krylov method for the delay eigenvalue problem*, SIAM J. Sci. Comput., 32 (2010), pp. 3278–3300. (Cited on pp. 50, 99, 103, 104, 106, 108, 110, 116, 119, and 127)

[50] E. Jarlebring, K. Meerbergen, and W. Michiels, *An Arnoldi method with structured starting vectors for the delay eigenvalue problem*, in 9th IFAC Workshop on Time Delay Systems, T. Vyhlidal and P. Zitek, eds., Prague, June 7-9, 2010. (Cited on pp. 9, 40, and 44)

[51] E. Jarlebring, K. Meerbergen, and W. Michiels, *Computing a partial Schur factorization of nonlinear eigenvalue problems using the Infinite Arnoldi method*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 411–436. (Cited on p. 92)

[52] E. Jarlebring, W. Michiels, and K. Meerbergen, *A linear eigenvalue algorithm for the nonlinear eigenvalue problem*, Numer. Math., 122 (2012), pp. 169–195. (Cited on pp. 9, 42, 43, 45, 59, 91, 92, 94, 95, 96, 101, and 125)

[53] K. Ko, N. Folwell, L. Ge, A. Guetz, L.-Q. Lee, Z. Li, C. Ng, E. Prudencio, G. Schussman, R. Uplenchwar, and L. Xiao, *Advances in electromagnetic modelling through high performance computing*, Phys. C, 441 (2006), pp. 258–262. (Cited on p. 8)

[54] D. Kressner, *A block Newton method for nonlinear eigenvalue problems*, Numer. Math., 114 (2009), pp. 355–372. (Cited on p. 9)

[55] D. Kressner and J. E. Roman, *Memory-efficient Arnoldi algorithms for linearizations of matrix polynomials in Chebyshev basis*, Numer. Linear Algebra Appl., 21 (2014), pp. 569–588. (Cited on pp. 120 and 128)

[56] V. N. Kublanovskaya, *On some algorithms for the solution of the complete eigenvalue problem*, USSR Comput. Math. Math. Phys., 1 (1962), pp. 637–657. (Cited on p. 3)

[57] V. N. Kublanovskaya, *The application of Newton's method to the determination of the eigenvalues of λ-matrices*, Dokl. Akad. Nauk SSR, 188 (1969), pp. 1004–1005. (Cited on p. 9)

[58] V. N. Kublanovskaya, *On an approach to the solution of the generalized latent value problem for λ-matrices*, SIAM J. Numer. Anal., 7 (1970), pp. 532–537. (Cited on p. 9)

[59] J. L. Lagrange, *Leçons élémentaires sur les mathématiques données à l'École Normale en 1795*, in Oeuvres de Lagrange VII, Gauthier-Villars, Paris, 1877, pp. 183–287. (Cited on p. 26)

[60] P. Lancaster, *A generalised Rayleigh quotient iteration for lambda-matrices*, Arch. Ration. Mech. Anal., 8 (1961), pp. 309–322. (Cited on p. 9)

[61] L.-Q. Lee, L. Ge, Z. Li, C. Ng, G. Schussman, and K. Ko, *Achievements in ISICs/SAPP collaborations for electromagnetic modeling of accelerators*, J. Phys.: Conf. Ser., 16 (2005), pp. 205–209. (Cited on p. 8)

[62] R. B. Lehoucq, *Analysis and Implementation of an Implicitly Restarted Arnoldi Iteration*, PhD thesis, Rice University, Houston, TX, 1995. (Cited on pp. 12, 15, 85, and 86)

[63] R. B. Lehoucq and D. C. Sorensen, *Deflation techniques for an implicitly restarted Arnoldi iteration*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 789–821. (Cited on pp. 3, 15, and 128)

[64] F. Leja, *Sur certaines suites liees aux ensembles plans et leur application a la representation conforme*, Ann. Polon. Math., 4 (1957), pp. 8–13. (Cited on pp. 56 and 58)

[65] C. S. Lent and D. J. Kirkner, *The quantum transmitting boundary method*, J. Appl. Phys., 67 (1990), pp. 6353–6359. (Cited on p. 6)

[66] A. L. Levin and E. B. Saff, *Optimal ray sequences of rational functions connected with the Zolotarev problem*, Constr. Approx., 10 (1994), pp. 235–273. (Cited on p. 76)

[67] A. L. Levin and E. B. Saff, *Potential Theoretic Tools in Polynomial and Rational Approximation*, in Harmonic Analysis and Rational Approximation, J.-D. Fournier, J. Grimm, J. Leblond, and J. R. Partington, eds., vol. 327 of Lecture Notes in Control and Information Science, Springer Berlin Heidelberg, Berlin, 2006, pp. 71–94. (Cited on p. 75)

[68] B.-S. Liao, Z. Bai, L.-Q. Lee, and K. Ko, *Nonlinear Rayleigh-Ritz iterative method for solving large scale nonlinear eigenvalue problems*, Taiwanese J. Math., 14 (2010), pp. 869–883. (Cited on pp. 8, 59, 81, and 134)

[69] D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann, *Vector spaces of linearizations for matrix polynomials*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 971–1004. (Cited on pp. 4, 24, 26, 116, and 119)

[70] D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann, *Smith forms of palindromic matrix polynomials*, Electron. J. Linear Algebra, 22 (2011), pp. 53–91. (Cited on p. 23)

[71] M. Martinez Aguirre, *Experimental and Numerical Dynamic Analysis of Press-Formed Viscoelastic Sandwich Structures*, PhD thesis, University of Mondragón, Mondragón, 2011. (Cited on pp. 5 and 138)

[72] K. Meerbergen, *The quadratic Arnoldi method for the solution of the quadratic eigenvalue problem*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1463–1482. (Cited on p. 3)

[73] V. Mehrmann and H. Voss, *Nonlinear eigenvalue problems: A challenge for modern eigenvalue methods*, GAMM-Mitt., 27 (2004), pp. 121–152. (Cited on p. 9)

[74] W. Michiels and S.-I. Niculescu, *Stability, Control, and Computation for Time-Delay Systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2 ed., 2014. (Cited on p. 4)

[75] R. B. Morgan, *On restarting the Arnoldi method for large nonsymmetric eigenvalue problems*, Math. Comp., 65 (1996), pp. 1213–1231. (Cited on pp. 15 and 128)

[76] Y. Nakatsukasa and F. Tisseur, *Eigenvector error bounds and perturbation for nonlinear eigenvalue problems*, in Manchester Workshop on Nonlinear Eigenvalue Problems (NEP14), Manchester, Apr. 2014. (Cited on pp. 116 and 119)

[77] S. Nazarian and M. R. Desai, *Automated surface wave method: Field testing*, J. Geotech. Eng., 119 (1993), pp. 1094–1111. (Cited on p. 7)

[78] A. Neumaier, *Residual inverse iteration for the nonlinear eigenvalue problem*, SIAM J. Numer. Anal., 22 (1985), pp. 914–923. (Cited on p. 9)

[79] G. Opitz, *Steigungsmatrizen*, ZAMM Z. Angew. Math. Mech., 44 (1964), pp. T52–T54. (Cited on pp. 36 and 37)

[80] M. D. Rao, *Recent applications of viscoelastic damping for noise control in automobiles and commercial airplanes*, J. Sound Vib., 262 (2003), pp. 457–474. (Cited on pp. 5 and 138)

[81] L. Reichel, *Newton interpolation at Leja points*, BIT, 30 (1990), pp. 332–346. (Cited on pp. 58, 67, and 77)

[82] A. Ruhe, *Algorithms for the nonlinear eigenvalue problem*, SIAM J. Numer. Anal., 10 (1973), pp. 674–689. (Cited on p. 9)

[83] A. Ruhe, *Rational Krylov sequence methods for eigenvalue computation*, Linear Algebra Appl., 58 (1984), pp. 391–405. (Cited on pp. 16 and 48)

[84] A. Ruhe, *Eigenvalue Algorithms with Several Factorizations - A Unified Theory yet?*, tech. report, Chalmers University of Technology, Göteborg, 1998. (Cited on p. 18)

[85] A. Ruhe, *Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils*, SIAM J. Sci. Comput., 19 (1998), pp. 1535–1551. (Cited on pp. 16, 19, 48, 55, and 128)

[86] Y. Saad, *Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices*, Linear Algebra Appl., 34 (1980), pp. 269–295. (Cited on p. 13)

[87] B. Sadiq and D. Viswanath, *Barycentric Hermite interpolation*, SIAM J. Sci. Comput., 35 (2013), pp. A1254–A1270. (Cited on p. 29)

[88] E. B. Saff and V. Totik, *Logarithmic Potentials with External Fields*, vol. 316, Springer Berlin Heidelberg, Berlin, 1997. (Cited on p. 75)

[89] M. Schevenels, *The Impact of Uncertain Dynamic Soil Characteristics on the Prediction of Ground Vibrations*, PhD thesis, K.U. Leuven, Leuven, 2007. (Cited on p. 146)

[90] M. Schevenels, S. François, and G. Degrande, *EDT: An elastodynamics toolbox for MATLAB*, Comput. Geosci., 35 (2009), pp. 1752–1754. (Cited on p. 146)

[91] M. Schevenels, S. François, and G. Degrande, *EDT: ElastoDynamics Toolbox for Matlab*, tech. report, Department of Civil Engineering, K.U. Leuven, Leuven, 2010. (Cited on p. 146)

[92] C. Schneider and W. Werner, *Hermite interpolation: The barycentric approach*, Computing, 46 (1991), pp. 35–51. (Cited on p. 29)

[93] A. Shakoori, *Bivariate Polynomial Solver by Values*, PhD thesis, University of Western Ontario, London, Ontario, 2007. (Cited on p. 33)

[94] G. L. G. Sleijpen and H. A. Van der Vorst, *A Jacobi–Davidson iteration method for linear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 401–425. (Cited on p. 55)

[95] G. L. G. Sleijpen and H. A. Van der Vorst, *A Jacobi–Davidson iteration method for linear eigenvalue problems*, SIAM Rev., 42 (2000), pp. 267–293. (Cited on p. 55)

[96] D. C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385. (Cited on p. 13)

[97] G. W. Stewart, *A Krylov–Schur algorithm for large eigenproblems*, SIAM J. Matrix Anal. Appl., 23 (2002), pp. 601–614. (Cited on pp. 13 and 15)

[98] Y. Su and Z. Bai, *Solving rational eigenvalue problems via linearization*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 201–216. (Cited on pp. 4 and 88)

[99] Y. Su, J. Zhang, and Z. Bai, *A compact Arnoldi algorithm for polynomial eigenvalue problems*, in Recent Advances in Numerical Methods for Eigenvalue Problems (RANMEP2008), Taiwan, Jan. 2008. (Cited on p. 120)

[100] F. Tisseur and K. Meerbergen, *The quadratic eigenvalue problem*, SIAM Rev., 43 (2001), pp. 235–286. (Cited on p. 3)

[101] A. Townsend, V. Noferini, and Y. Nakatsukasa, *Vector Spaces of Linearizations for Matrix Polynomials: A Bivariate Polynomial Approach*, tech. report, The University of Manchester, Manchester, 2012. (Cited on p. 35)

[102] L. N. Trefethen, *Spectral Methods in MatLab*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2000. (Cited on p. 103)

[103] M. Van Barel, *Designing rational filter functions for solving eigenvalue problems by contour integration*, Tech. Report TW657, Department of Computer Science, KU Leuven, Leuven, 2015. (Cited on pp. 9 and 157)

[104] M. Van Barel and P. Kravanja, *Nonlinear Eigenvalue Problems and Contour Integrals*, Tech. Report TW656, Department of Computer Science, KU Leuven, Leuven, 2014. (Cited on p. 9)

[105] R. Van Beeumen, E. Jarlebring, and W. Michiels, *A Rank-Exploiting Infinite Arnoldi Algorithm for Nonlinear Eigenvalue Problems*,

Tech. Report TW652, Department of Computer Science, KU Leuven, Leuven, 2014. (Cited on pp. 20, 98, 108, and 110)

[106] R. Van Beeumen, K. Meerbergen, and W. Michiels, *A rational Krylov method based on Hermite interpolation for nonlinear eigenvalue problems*, SIAM J. Sci. Comput., 35 (2013), pp. A327–A350. (Cited on pp. 20, 46, 50, 51, 53, 80, 125, 127, 130, and 138)

[107] R. Van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for nonlinear eigenvalue problems*, SIAM J. Matrix Anal. Appl., (2015). (Cited on pp. 20, 56, 116, and 131)

[108] R. Van Beeumen, W. Michiels, and K. Meerbergen, *Linearization of Lagrange and Hermite interpolating matrix polynomials*, IMA J. Numer. Anal., (2014). (Cited on pp. 20, 30, 32, 33, 116, and 119)

[109] W. G. Vandenberghe, M. V. Fischetti, R. Van Beeumen, K. Meerbergen, W. Michiels, and C. Effenberger, *Determining bound states in a semiconductor device with contacts using a nonlinear eigenvalue solver*, J. Comput. Electron., 13 (2014), pp. 753–762. (Cited on pp. 6, 21, 86, 140, and 144)

[110] D. Verhees, R. Van Beeumen, K. Meerbergen, N. Guglielmi, and W. Michiels, *Fast algorithms for computing the distance to instability of nonlinear eigenvalue problems, with application to time-delay systems*, Int. J. Dynam. Control, 2 (2014), pp. 133–142. (Cited on p. 156)

[111] R. von Mises and H. Pollaczek-Geiringer, *Praktische Verfahren der Gleichungsauflösung*, ZAMM Z. Angew. Math. Mech., 9 (1929), pp. 152–164. (Cited on p. 3)

[112] H. Voss, *An Arnoldi method for nonlinear eigenvalue problems*, BIT, 44 (2004), pp. 387–401. (Cited on p. 9)

[113] H. Voss, *A Jacobi–Davidson method for nonlinear and nonsymmetric eigenproblems*, Comput. Struct., 85 (2007), pp. 1284–1292. (Cited on p. 9)

[114] H. Voss, *Nonlinear Eigenvalue Problems*, in Handbook of Linear Algebra, Taylor & Francis, 2 ed., 2013, ch. 60, pp. 60.1–60.24. (Cited on p. 9)

[115] J. L. Walsh, *On interpolation and approximation by rational functions with preassigned poles*, Trans. Amer. Math. Soc., 34 (1932), pp. 22–74. (Cited on pp. 75 and 76)

[116] J. L. Walsh, *Interpolation and approximation by rational functions in the complex domain*, American Mathematical Society, Providence, 5 ed., 1969. (Cited on pp. 75 and 76)

[117] J. L. Walsh and H. G. Russell, *Hyperbolic capacity and interpolating rational functions II*, Duke Math. J., 33 (1966), pp. 275–279. (Cited on p. 75)

[118] E. Waring, *Problems concerning interpolations*, Philos. Trans. R. Soc. Lond., 69 (1779), pp. 59–67. (Cited on p. 26)

[119] J. Wu, *Theory and Applications of Partial Functional Differential Equations*, vol. 119 of Applied Mathematical Sciences, Springer, New York, NY, 1996. (Cited on p. 110)

[120] Z. Wu and W. Michiels, *Reliably computing all characteristic roots of delay differential equations in a given right half plane using a spectral method*, J. Comput. Appl. Math., 236 (2012), pp. 2499–2514. (Cited on pp. 110 and 113)

[121] S. Yokota and T. Sakurai, *A projection method for nonlinear eigenvalue problems using contour integrals*, JSIAM Lett., 5 (2013), pp. 41–44. (Cited on p. 9)

[122] D. Yuan and S. Nazarian, *Automated surface wave method: Inversion technique*, J. Geotech. Eng., 119 (1993), pp. 1112–1126. (Cited on p. 7)

[123] Y. Zhang and Y. Su, *A memory-efficient model order reduction for time-delay systems*, BIT, 53 (2013), pp. 1047–1073. (Cited on p. 120)

Curriculum Vitae

General

Name:	Roel Van Beeumen
Date of birth:	24 April 1987
Place of birth:	Wilrijk (Antwerp)
E-mail:	Roel.VanBeeumen@cs.kuleuven.be
URL:	`http://people.cs.kuleuven.be/~roel.vanbeeumen`

Experience

- **PhD Student** 2011-2015
  Department of Computer Science, KU Leuven, Belgium.

- **Teaching Assistant** 2010–2011
  Department of Computer Science, KU Leuven, Belgium.

Education

- **Master of Arts in Archaeology** 2010–2011
  KU Leuven, Belgium.

- **Master of Engineering: Mathematical Engineering** 2008–2010
  KU Leuven, Belgium.

- **Bachelor of Archaeology** 2006–2010
  KU Leuven, Belgium.

- **Bachelor of Engineering: Mechanical Engineering** 2005–2008
  KU Leuven, Belgium.

List of Publications

International journal papers

[8] R. Van Beeumen, E. Jarlebring, and W. Michiels, *A rank-exploiting infinite Arnoldi algorithm for nonlinear eigenvalue problems*, Numerical Linear Algebra with Applications, in revision.

[7] R. Van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for nonlinear eigenvalue problems*, SIAM Journal on Matrix Analysis and Applications, (2015), accepted.

[6] R. Van Beeumen, W. Michiels, and K. Meerbergen, *Linearization of Lagrange and Hermite interpolating matrix polynomials*, IMA Journal of Numerical Analysis, (2014), published online.

[5] S. Güttel, R. Van Beeumen, K. Meerbergen, and W. Michiels, *NLEIGS: A class of fully rational Krylov methods for nonlinear eigenvalue problems*, SIAM Journal on Scientific Computing, 36 (2014), pp. A2842–A2864.

[4] W.G. Vandenberghe, M.V. Fischetti, R. Van Beeumen, K. Meerbergen, W. Michiels, and C. Effenberger, *Determining bound states in a semiconductor device with contacts using a nonlinear eigenvalue solver*, Journal of Computational Electronics, 13 (2014), pp. 753–762.

[3] D. Verhees, R. Van Beeumen, K. Meerbergen, N. Guglielmi and W. Michiels, *Fast algorithms for computing the distance to instability of nonlinear eigenvalue problems, with application to time-delay systems*, International Journal of Dynamics and Control, 2 (2014), pp. 133–142.

[2] R. Van Beeumen, K. Meerbergen, and W. Michiels, *A rational Krylov method based on Hermite interpolation for nonlinear eigenvalue problems*, SIAM Journal on Scientific Computing, 35 (2013), pp. A327–A350.

[1] R. Van Beeumen, K. Van Nimmen, G. Lombaert, and K. Meerbergen, *Model reduction for dynamical systems with quadratic output*, International Journal for Numerical Methods in Engineering, 91 (2012), pp. 229–248.

International conferences proceedings – full papers

[2] R. Van Beeumen, P. Degryse, M. Eisenga, and E. Muijsenberg, *Numerical modelling of ancient primary glass furnaces*, in Proceedings of the 11th International Seminar on Furnace Design Operation & Process Simulation, J. Chmelař, E. Muijsenberg, and L. Němec, eds., Velké Karlovice, June 2011, Glass Service, Inc., pp. 141–149.

[1] R. Van Beeumen and K. Meerbergen, *Model Reduction by Balanced Truncation of Linear Systems with a Quadratic Output*, AIP Conference Proceedings, 1281 (2010), pp. 2033–2036.

International conferences proceedings – abstracts

[9] R. Van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for solving nonlinear eigenvalue problems*, in IMA Conference on Numerical Linear Algebra and Optimisation, Birmingham, Sept. 2014.

[8] R. Van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for solving nonlinear eigenvalue problems*, in SIAM Annual Meeting, Chicago, IL, July 2014.

[7] R. Van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for the nonlinear eigenvalue problem*, in Householder Symposium XIX, Spa, June 2014.

[6] R. Van Beeumen, K. Meerbergen, and W. Michiels, *Compact rational Krylov methods for solving nonlinear eigenvalue problems*, in Manchester Workshop on Nonlinear Eigenvalue Problems, Manchester, Apr. 2014.

[5] R. Van Beeumen, S. Güttel, K. Meerbergen, and W. Michiels, *NLEIGS: A class of robust fully rational Krylov methods for nonlinear eigenvalue problems*, in GAMM Workshop on Applied and Numerical Linear Algebra, Wuppertal, Sept. 2013.

[4] R. Van Beeumen, K. Meerbergen, and W. Michiels, *A practical rational Krylov algorithm for solving large-scale nonlinear eigenvalue problems*, in Conference of the International Linear Algebra Society (ILAS2013), Providence, RI, June 2013.

[3] R. Van Beeumen, K. Meerbergen, and W. Michiels, *A rational Krylov method based on Hermite interpolation for the nonlinear eigenvalue problem*, in International Congress on Computational and Applied Mathematics (ICCAM2012), Ghent, July 2012.

[2] R. Van Beeumen, K. Meerbergen, and W. Michiels, *A rational Krylov method based on Newton and/or Hermite interpolation for the nonlinear eigenvalue problem*, in SIAM Conference on Applied Linear Algebra, Valencia, June 2012.

[1] R. Van Beeumen and K. Meerbergen, *Model reduction by balanced truncation of linear systems with a quadratic output*, in International Congress on Computational and Applied Mathematics (ICCAM2010), Leuven, July 2010.

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
TWR GROUP & NALAG GROUP
Celestijnenlaan 200A box 2402
B-3001 Leuven
Roel.VanBeeumen@cs.kuleuven.be
http://people.cs.kuleuven.be/~roel.vanbeeumen